Introduction

Corporate IT is in a state of crisis today. Too many demands on too complex infrastructures with cost containment and productivity enhancement pressures ever increasing. IT has responded to this state with grit, determination and hard work. Yet because there are so many demands that carry so much pressure, IT hasn’t put effective infrastructure or processes in place to tackle this crisis.

Application infrastructure configuration management is a core discipline necessary to effectively manage today’s IT infrastructure. Highly-effective IT organizations often practice a common set of configuration management processes, and that’s what this paper is about. The processes, policies and procedures that very-effective IT organizations use to manage application infrastructures. These represent the best practices in application infrastructure configuration management.

Best practices are simply those activities that have proven successful in other organizations like yours. Many of them will be similarly applicable to your environment while some may not be appropriate.
IT infrastructure is a broad term, potentially encompassing interrelated and dependent components from network switches to application and web servers. In this paper, we use the term application infrastructure to mean the set of infrastructure components above the operating system such as databases, messaging servers, application servers and web servers. Application infrastructure configuration management is the discipline of creating and making changes to these application components. Like the related, but different discipline software configuration management, application infrastructure configuration management focuses on the processes for and policies about making changes to infrastructure components that support enterprise applications.

Your organization likely practices application infrastructure configuration management if it runs any of the infrastructure layers including databases, web servers or application servers. When an Oracle WebLogic administrator makes changes to a WebLogic application server, he is practicing application infrastructure configuration management.

Characteristics of Effective Application Infrastructure Configuration Management Processes

Most organizations put configuration change management in place to avoid future problems or remedy existing problems. Infrastructure configuration management problems surface in three areas: unplanned downtime, application and/or change backlog and risk management.

- **Unplanned downtime** is the most visible, most easily measured and often the area in which organizations base pay for performance scales.

- **Application or change backlog** is usually the result of rigid and inflexible change processes or staff overload. It is a natural consequence of a heavy focus on unplanned downtime – the easiest way to reduce unplanned downtime is to limit the number of changes introduced.

- **Risk management** is the least understood and most poorly defined area, and usually only surfaces as a result of external requirements (such as regulatory compliance) or as a side effect of unplanned downtime or application backlog (if you don’t understand your environment well, it’s difficult to make changes that don’t affect uptime).

Effective IT organizations have above industry-norm metrics for these areas. For example, effective organizations measure, refine and improve:

1. Their change success rate overall and their change success rate by component. That is, the % of successful changes, versus those that aren’t.

2. Their response turnaround to change requests. Most organizations categorize changes according to severity and top-performing organizations measure, refine and improve their turnaround by change request severity.

3. The quantity and distribution of unauthorized changes (that is, changes made outside the change process).

4. Their backlog of projects requested by external stakeholders.

5. Their frequency and quantity of unplanned down events. The sources of unplanned down events by component and root cause.

6. Their mean time to repair unplanned down events.

Best Practices

1. Define a Change Process

One of the simplest ways to reduce downtime is to stop introducing changes altogether in an infrastructure. But for most organizations this isn’t realistic, particularly in today’s J2EE environments. Change is an integral and necessary part of application infrastructure configuration management. Infrastructure teams are better off expecting change and putting solid processes in place to manage it rather than resisting it and getting surprised by the consequences. Defining a change process that works for your organization, training users on it, putting appropriate tools in place to make it effective, efficient and enforced is one of the most effective best practices in this area.
An effective change process includes five main components:

- **Change policy**: the procedures, processes and rules users must follow to implement a change. It’s important to write these to accommodate both normal and emergency changes, because both are likely. If the policy doesn’t accommodate emergency situations then the team either cannot respond appropriately to emergencies, or must go around the policy to implement emergency fixes. Some organizations err on the side of a more flexible change policy in the belief that it makes them more responsive. Organizations that write policies that are more specific are usually more effective, because these organizations have analyzed their change processes and defined specific actions for different situations.

- **User education**: users must know how to follow the process, and the consequences of not following it.

- **Change review board**: this isn’t necessarily the same group of people in all cases, but the policy must indicate who must review and approve requests for changes.

- **Rollback facilities**: Your change policy can prescribe which types of changes require rollback plans, although best practices indicate that every change should include a rollback plan or facility. This can seem overly burdensome for simple changes, although change management tools, such as Application Configuration Console can automatically provide this capability.

- **Centralized change**: While you can’t realistically stop changes altogether, you can limit when and how they occur. Most organizations define change windows in their change policy, such as once a week in a specified time period. You should also limit who can make changes, both through your policy and through access controls. Finally, you should limit where these individuals can make changes (see 2. Centralize Application Infrastructure Configuration Management below) for more information on this topic.

An effective change process doesn’t necessarily require an Application infrastructure configuration management solution, such as Application Configuration Console. However this solution can significantly accelerate a change process implementation, improve compliance and accuracy, and augment necessary metrics collection and reporting.

### 2. Centralize Application Infrastructure Configuration Management

By ‘centralize’ we mean to get all configuration items in a centrally accessible database, which users, reviewers and managers can access. The best practice includes more than simply collecting configuration items into it. It includes actively managing infrastructure from this central database – making changes to the database, reviewing them in the context of the rest of the items in the database, and implementing changes by pushing the contents of the database out to the live infrastructure.

Before commercial application infrastructure configuration management tools such as Application Configuration Console, IT organizations did one of three things to manage configurations of application infrastructure – built tools themselves, attempted to adapt software configuration management tools to the task or did nothing. If your organization is in the third camp and does nothing, you are not alone. By far the most predominant approach is to do nothing. Infrastructure specialists, such as application server administrators, simply make changes to the application infrastructure in place and hope for the best.

This minimalist approach can be effective if you have a highly trained and disciplined staff and a low volume of changes. But it has disadvantages including:

- You have no backup of the running systems. If servers crash you have no way to recover the configuration state.

- You have no audit trail of changes, other than server logs, so you have no way of knowing what changes were made, when, why and by whom.

- You have no automatic rollback. If you follow the best practice of putting a rollback plan in place for every change, you place an additional burden on every change author to define and implement a rollback facility.

- It’s very difficult for stakeholders and reviewers to understand a change in the context of the overall system.

- You usually lack fine-grained controls over who can change what, and when.

By implementing a centralized database of configuration items such as in Application Configuration Console, you get distinct
Application Configuration Console Best Practices

advantages, including:

- A central store of configuration items separate from the ‘as built’ systems to facilitate modeling, backup and change management.

- Centralized and automatic audit trails of changes. These include the who, what, when and why of all changes.

- Automatic rollback facilities. This means that change initiators need not spend time on designing a rollback facility - it is an automatic part of implementing the change.

- Complete version histories of changes, which aid in troubleshooting and metrics collection.

- Fine-grained permissions to configuration items within the database meaning that you can provide narrowly-defined permissions to change different items to different stakeholders.

3. Track Infrastructure for Changes

Sadly, many infrastructure teams don’t fully control their environments. While they are the first ones to get a trouble call, they can’t mandate that changes to infrastructure only go through approved channels, or are approved by them. Tracking infrastructure for changes provides these teams two benefits. First, it provides early warning on potential issues. Second, it provides version histories which are vital to problem isolation and which improve mean time to repair as well as providing important metrics for focus area of improvement programs.

Tracking for changes is not a replacement for a solid and enforceable change management process. You should implement Application infrastructure change management and use tracking to augment it, particularly for detecting compliance with the process. The objective in tracking for early warning is to get appropriate information to appropriate owners, stakeholders and managers of changes to application infrastructure, so they can act before they possibly have to react to an incident.

The objective of tracking for change audit logging is to build a history of changes to infrastructure for the purposes of problem determination, metrics and regulatory and policy compliance.

This type of tracking is just about what changes have been implemented. Most changes follow the change process and have the desired effect. This tracking is intended to find the ones that don’t follow the process.

4. Create, Use and Reuse Gold Masters

A ‘gold master’ is a configuration that has been tested and known to work. Many IT teams ‘hand craft’ each application infrastructure for each new application and environment. Highly effective teams instead reuse gold master configurations for each new application and environment. In order to capture, catalog and reuse gold masters you must have an effective database of these configurations, which is one of several reasons for implementing the best practice of centralizing application infrastructure configuration management.

Replace Rather Than Fix

When you have a catalog of known-to-work gold masters, it is usually faster to replace the configurations in a broken component rather than attempting to fix it. Disciplined teams use a triage approach – they spend a fixed amount of time (two to four hours is typical) trying to understand and fix an issue. If unsuccessful, they replace the configurations in the component with gold masters.

Replace Run Book Processes with Gold Masters

Similar to the discussion in the previous section on reuse of configuration items, IT organizations can gain tremendous leverage by reusing previous successful processes. However, keep in mind that the objective of these processes is the result, not the repetitive application of the process. For example, you might have defined ‘run book’ processes for how you provision a new application server. Your objective in these processes is not to do the same steps each time you need a new application server. Your objective is to get the same results. That is, an application server that performs the same way as other application servers that you know work. You document and repeat the steps because you know these steps result in a working server.

When you implement a facility for creating, cataloging and using gold master configurations such as Application Configuration Console, you have a new facility to augment these processes. You capture the ‘as-built’ system in the gold
Application Configuration Console Best Practices

You can then apply these configuration settings to a new, as yet un-provisioned environment. The result is, as before, a running environment (or server in the above example), but you have avoided the process steps you followed to get to the original environment.

5. Collect, Review and Improve Key Metrics

Key metrics for application infrastructure configuration management are those measures of an effective IT organization discussed in the section above Characteristics of Effective Infrastructure Configuration Management Processes. They include the obvious ones that you likely already measure like downtime and backlog. However effective organizations collect additional metrics for the number of changes and their success rate, the staff time spent on planned versus unplanned work, and the number and percent of unauthorized changes.

Metrics collection, reporting and analysis is another area where application infrastructure configuration management tools such as Application Configuration Console can play a large role. Without automated collection of these metrics, most organizations simply don’t have the time to retroactively collect and analyze them. On the other hand, Application Configuration Console automatically collects these key metrics and provides reporting for analysis.

7. Automate Repetitive and Redundant Tasks

Even after fully implementing the best practices described above, IT teams spend a lot of time in repetitive tasks that are likely error-prone. This best practice suggests that you analyze the activities of your IT team and automate those tasks that they perform repeatedly. Ideally you would automate all tasks, but most IT teams assignments are too diverse to reach this goal. Instead you should go after the tasks that have the biggest payoff in terms of improved quality of operations and throughput. This usually means attacking the ones that you do repeatedly, not the ones that are the hardest.

You can automate almost any task, but the tradeoff is in the effort to implement and support the automation. Unless you have a platform on which to implement automations, usually built on a centralized database, the support burden can quickly outweigh the advantages of the automation in the first place. This is the reason many IT teams don’t automate more, and a key advantage of implementing an application infrastructure configuration management solution like Application Configuration Console.

Summary

Wherever your IT organization lands on a spectrum from chaos to a highly-effective, accurate and responsive team, best practices can guide you in improvement. No doubt you already employ some of the practices mentioned in this document and others aren’t appropriate to your environment. As with any process guidance, think of these as recommendations, not requirements. Apply the ones that make sense for your organization as they are sure to lead to more efficient operations with higher quality and throughput.

About Application Configuration Console

IT teams use Application Configuration Console to manage changes to infrastructure configurations. Application Configuration Console includes key capabilities in the areas of application infrastructure configuration management including:

• Centralize infrastructure configuration items
• Track infrastructure for changes
• Create, use and reuse gold masters
• Collect and report on key metrics
• Automate repetitive tasks

Application Configuration Console provides these capabilities on a foundation for an effective and high-performance change management process.
Application Configuration Console Best Practices