Operational Plan Development for Application A

Purpose:
This document defines the proposed operational plan for Application A, to ensure our output and deliverables follow a consistent and repeatable process that promotes communication, quality, productivity, and efficiency. It also identifies plan to use key service level metrics and measures to gather health and performance metric for service improvements while minimizing potential disruption or outages of business services for our business partners.

1. SERVICE LEVEL MANAGEMENT

The goals of Service Level Management and reporting include:

- Provide an accurate overview of an application in terms of incidents, problems and availability
- Provide an indicator of product health or service trends
- Report generation is automatic eliminating manual compiling
- On demand access to execute a report for a desired reporting period
- On demand request for all or a subset of reports for a desired period
- Rolling three month window to illustrate short term trends
- Report data is based on standard business hours
- Scheduled maintenance or downtime is excluded from all reports

Approved reports: Reports are considered a Controlled Item and should follow formal change management processes to ensure changes are tested and verified as producing correct data. New reports may be requested at any time. Annual review of reports and report data to ensure data provided is valuable to the business.

Recommended Tool: Jira

Critical Success Factors: These are the identified critical factors to ensure accurate reports of Application A. It is critical that all incidents be consistently entered and updated in Jira. This consistency is required to provide the source data for Incident, Problem, Maintenance and service.

- Jira will be use as the source data for all incidents and problems for Application A
- All support personnel are diligent in the use of JIRA to ensure current and accurate data
- Tickets are created to capture all work performed
- Ticket status updated immediately when working a ticket
- Ticket logs updated to identify resolution and the product or service that was the root cause of the issue.

Service level management identifies metrics that are a truthful reflection of the customer's actual experiences and level of satisfaction. This can involve collating and analysing information from:

- **Customer engagement**: Involves initial listening, discovery and information capture on which to base metrics, measures and ongoing progress discussion.
- **Customer feedback**: This ideally gathered from a number of sources such as surveys that could be event based that requires immediate feedback or from a more reflective periodic survey.
- **Operational metrics**: These are low level indicators of various operational activities including system availability, incident response and fix times. In this case, the system availability is a high-level indicator as this is an active production environment with a high user load and a low tolerance for downtime.
- **Business metrics**: This can be any business activity that is important or valuable by the customer and used as a means of gauging the success of the service.

Once this feedback is gathered and collated for ongoing review, it can be used as input to design suitable measurement and reporting models and practices.

2. INCIDENT MANAGEMENT:

This describes the process to open and resolve incidents. An Incident is an event that disrupts, or could disrupt, applications and services within the scope of Application A. Incidents should be reported by end users through the Jira Service Desk or can be based on technical resource observations about system conditions during health checks.

Recommended Tool: Jira

Objective and Goals

Incident Management objectives include:

- Log and manage incidents
- Priority based on business impact
- Escalation
- Resolution and Closure
- Managed through toolsets to give access to service level, configuration and knowledge data

The goals of Incident Management include:

- Minimize the negative impact of incidents by prompt detection and resolution of incidents
- Identification of potential improvements to services
- Identification of additional service or business training requirements

Issues with Application A within the production environment can trigger an Incident. The overall objective of Incident Management is restoration of service for the affected business users and areas as soon as possible either by resolving the root cause or by providing a temporary fix or workaround. Incident management can have an enormous impact on customer and user satisfaction, and how they perceive the service provider.

3. MONITORING AND EVENT MANAGEMENT

This process is to systematically observe services and service components, and document and report selected changes of state identified as events. An event can be described as any change of state that has significance effect on IT services or components.

This practice identifies and priorities infrastructure, services, business processes and information security events. It establishes the appropriate response to the events including responding to conditions that could lead to potential faults or incidents. Events are identified most of the time through notifications created by a monitoring tool.

Monitoring and event management practice manages events to prevent, minimize and eliminate their negative impacts on business. It provides early detection to conditions that could have potential significance. The process also includes documentation and managing the monitored changes that are defined to determine their significance, identify, and initiate the correct control action to manage them.

Automation is key to successful monitoring and event management. Some services components come equipped with built-in monitoring and reporting capabilities that can be configured to meet the need of the practice. It is sometimes important to implement and configure purpose-built monitoring tools.

Metrics, monitoring, and alerting are all interrelated concepts that together form the basis of a monitoring system. They can provide visibility into the health of systems, help you understand trends in usage or behavior, and to understand the impact of changes you make.

List of core metrics/signals that should be collected in the production environment:

Host-Based Metrics:

These metrics would be anything involved in evaluating the health or performance of an individual machine which can include data about disk space, CPU load, swap usage, processes. are already available, provide value immediately, and can be forwarded to a monitoring system without much additional work. Many web servers, database servers, and other software also provide their own metrics which can be passed forward as well.

Application Metrics

These metrics help to determine whether an application is functioning correctly and with efficiency. Metrics at this level are indicators of the health, performance, or load of an application:

- Error and success rates
- Performance and latency of responses
- Service failures and restarts
- Resource usage

Network and Connectivity Metrics

For most types of infrastructure, network and connectivity indicators will be another dataset worth exploring. These are important gauges of outward-facing availability but are also essential in ensuring that services are accessible to other machines for any systems that span more than one machine. Networks should be checked for their overall functional correctness and their ability to deliver necessary performance by looking at:

- Connectivity
- Error rates and packet loss
- Latency
- Bandwidth utilization

This monitoring can help improve the availability and responsiveness of both your internal and external services.

Server Pool Metrics

When dealing with horizontally scaled infrastructure, another layer of infrastructure you will need to add metrics for is pools of servers. While metrics about individual servers are useful, at scale a service is better represented as the ability of a collection of machines to perform work and respond adequately to requests. This type of metric is in many ways just a higher-level extrapolation of application and server metrics, but the resources in this case are homogeneous servers instead of machine-level components. Some data you might want to track are:

- Pooled resource usage
- Scaling adjustment indicators
- Degraded instances

Collecting data that summarizes the health of collections of servers is important for understanding the actual capabilities of your system to handle load and respond to changes.

External Dependency Metrics

Often, services provide status pages or an API to discover service outages but tracking these within our systems as well as your actual interactions with the service can help identify problems with service providers that may affect operations. Some items that might be applicable to track at this level are:

- Service status and availability
- Success and error rates
- Run rate and operational costs
- Resource exhaustion

4. PROBLEM MANAGEMENT

A Problem is the unknown cause of one or more incidents or the underlying potential for causing an incident. This process will perform Problem Management, outlining the lifecycle of problems from root cause analysis to determining the long-term solutions to identified problems.
The process will also describe how descriptions of known errors and workarounds will be made available to the stakeholders. ITIL defines a known error as a problem with a documented root cause and a workaround.

Objective and Goals

The objectives of the Problem Management process are to:

- Prevent problems and resulting incidents from happening
- Eliminate recurring incidents
- Minimize the impact of incidents that cannot be prevented.

The goals of Problem Management include:

- Improve availability by enabling staff to speed up time to resolution
- Over time reducing the number and duration of incidents
- Reducing unplanned labor caused by incidents

Critical success factors for Problem Management are:

- Improved service quality
- Minimizing the impact of problems

Key performance indicators linked to these critical success factors (CSFs) are:

- Requests for changes that reduce the number of incidents
- Reduced number of incidents associated with previous months' top 5 problems
- Reduced backlog of problems for analysis

The Problem Management process covers the activities required to identify problems, maintain information about problems, including any workarounds or resolutions, perform root cause analysis and identify the problem's root cause. It also ensures the problem resolution is implemented following the appropriate control processes, including Change Management and Release Management. Problem resolution may involve application changes, but also changes to internal processes and procedures.

The Problem Management process begins when a problem is identified. Most frequently, the process will be triggered as an outcome of the Incident Management process, when recurring incidents are grouped together, and a Problem ticket is raised.
However, Problems may be identified at any time, even where recurring incidents have not been identified. For example, a Problem may be an outcome of Event Management, a code review, an application audit, or an identified performance issue.

The following requirements apply to triggering this process:

- Problem impact can be identified
- Problem is reproducible

5. SERVICE REQUEST REPORTING

The reports will provide operational metrics for Incidents and Problems. The reports are highly dependent on accuracy and the data in JIRA entered by users when working on tickets. Report design and execution should allow the report requestor to run all or a subset of the reports. All report data is derived from the JIRA projects.

Objective and Goals

The goals of Service Management Reporting include:

- Provide an accurate overview of an application in terms of incidents, problems and availability
- Provide an indicator of product health or service trends
- Report generation is automatic eliminating manual compiling.
- On demand access to execute a report for a desired reporting period
- On demand request for all or a subset of reports for a desired period
- Rolling three month window to illustrate short term trends
- Report data is based on standard business hours.
- Scheduled maintenance or downtime is excluded from all reports

Approved reports: Reports are considered a Controlled Item and will follow formal change management processes to ensure changes are tested and verified as producing correct data. New reports may be requested at any time and annual review of reports and report data to ensure data provided is valuable to the business.

Critical Success Factors

The following are critical factors to ensure accurate reports It is critical that all incidents be consistently entered and updated in JIRA. This consistency is required to provide the source data for Incident, Problem, maintenance and service JIRA AMS project is the source data for all incidents and problems

- All support personnel are diligent in the use of JIRA to ensure current and accurate data
- Tickets are created to capture all work performed
- Ticket status updated immediately when working a ticket
- Ticket logs updated to identify resolution and the product or service that that was the root cause of the issue
- Reports can pull data from multiple JIRA projects

6. CHANGE MANAGEMENT

This process is to make an initial and changed services and features available through ongoing development, maintenance, and support. A release may comprise many different infrastructure and application components that work together to deliver new or changed functionality. It may also include documentations, updated processes, or tools.

Objective and Goals

The goals of and objectives include:

- Release – To make available for use a version of the application, service or a collection of configuration items
- Release schedule – To document the timing of the release

This also includes:

- Continuously improve, document, and manage the technical Change Management (RFC) process ensuring strategic alignment using ITIL best practices
- Establish measures and metrics for monitoring change effectiveness
- Review and monitor releases and infrastructure change, including governance of schedules considering business priorities and enable change with minimal disruption.
- Ensure all change is visible to all stakeholders and communicated appropriately and all change risk is understood
- Develop a broad understanding of Applications A technical interfaces/functions with internal and external systems for contributing to risk assessment and change approvals
- Develop effective relationships with all stakeholders to support technical change by contributing to effective knowledge management practises

Critical Success Factors

- All aspects of governance are agreed upon mutually by release management and business area
- Fully accountable decision-making process
- Effective and efficient management committees

Key Performance Indicators

- Committee meetings held at all required intervals
- Record and post minutes for each committee meeting to SharePoint, Confluence or other agreed upon location
- Action items are logged with response plans identified; all action items are addressed within prescribed dates

Release can range in size; it may involve just a minor changed feature to a very large change involving many components that delivers a completely new service. In either case, a release plan will specify the exact combination of new changed components to be made available and the timing for the release through Jira Request for Change (RFC). The detailed worked in the release to each environment (Delivery, Test and Production) will be documented through Request for Deployment (RFD/RFD-Subtasks). A release schedule is used to document the timing for releases. This scheduled should be negotiated and agreed with all stakeholders. A release post implementation review enables learning and improvement opportunities and helps to ensure that customers users are satisfied.

Usually, almost all the release management work happens before deployment, with plans in place as to exactly which component will be deployed in that release with roll back plan.

Recommended approach to performing an upgrade of Application A:

- Change Requestor: Request the release of Application A by creating, completing and submission a Request for Change (RFC) which self-assess priority and risk of change. It is essential that the request provide enough information for visibility into exactly which changes are needed to which environments when, and by whom, and to be able to trace them back to a change set in version control. The change requestor provides an overall business approval to proceed with the major change.

- Change Manager: Review the RFC and approval if it is a standard change or request further review by Change Advisory Board (CAB) for non-standard change. If approved

- Schedule the change: The requestor work with the change manager to forward schedule the change tentatively

- Notify stakeholders: Inform all stakeholder about this major change by sending appropriate notices

- Request for Deployment (RFD): The change requestor or change coordinator create and submit RFD for each environment starting from the lower environment to production. This includes working with the change manager to plan and confirm the readiness of the release, resource availability and the deployment plan including to implement, verify, accept or rollback. They will be working with the change coordinator to firm up release dates. The RFD process will detail the implementation steps as the release moves through environments and generate appropriate communications according to the Operations Communication Plan.

  - The RFD process will ensure the RFC proposed dates are updated with any changes as the release progresses and communications are updated.
  - The RFD process will expose visibility of changes that will require outages/downtime.
  - Change may be performed by one or more members of the full Deliveries, DBA and Middle-Tier teams OR a contracted Vendor team.
  - Change Coordinator will coordinate a rollback in consultation with the Release Manager if significant issues are discovered after the release and/or a rollback is requested by the Change Requestor or Change Sponsor.
  - The rollback plan will have been built into the RFD process during Planning.
  - The Change Requestor may decide to accept the change with the known issues and move further remediation to a future RFC. Known issues that could result in Service Desk reports will be documented with the Service Desk
    - If a rollback is performed, the RFC must be automatically flagged for a post-implementation review (PIR) with the CAB. The RFD process may also flag the RFC for a post-implementation review for other concerns during deployment to be reviewed (for recommendations for improvement). Rollbacks will clear the RFC proposed dates as necessary
- The RFD process may update the RFC indicating the "deployed to" environment information automatically for all successful or accepted deployments
- Deployments to environments other than Production are redirected back to the Change Coordinator to plan additional deployments to subsequent environments with the Release Manager
- Deployments to Production are directed back to the Change Manager to close off the RFC and update the deployed-in environments after UAT sign off by the business area
- RFD closure may be automated to trigger when all rfd subtask are closed

Further recommendation:

- To deploy to this active production environment with a high user load and a low tolerance for downtime, I will recommend Production deployment is carried out outside regular business hours (late in the evening on weekdays or over the weekend) to reduce service disruption. Overtime planning and approval will be required.

- Provide outage notice way ahead of time to give stakeholders and users enough room to prepare.