A Global Operational Readiness Review Process: Improving Cloud Availability

James J. Cusick
Ritsumeikan University
Kusatsu, Shiga, Japan
j.cusick@computer.org

Lija Basil
HCL Tech
Tamil Nadu, India
lija.subin@gmail.com

Abstract—The ORR (Operational Readiness Review) is a long-standing practice to help ensure application or system readiness and improved Availability. In this paper the ORR is defined and recent examples of its use from Cloud Computing environments are compared. An emphasis on ORRs used within DevOps environments is also provided. A detailed presentation of a custom ORR implementation for a large global IT organization is shared. This includes the process development approach, key components of the ORR checklist, automation support provided, and a unique Executive dashboard visualization of status on in-flight releases. Challenges and benefits from this ORR implementation are provided as well as a discussion of industry checklists and their associated Production Readiness Reviews (PRR)/ORR. Finally, suggestions for further improvements, automation, and usage of the ORR in large-scale industrial settings based on this real-world experience are elaborated.

Keywords—operational readiness review, availability, site reliability engineering, release management.

I. INTRODUCTION

One can design-in quality, attempt to test-in quality, or fix applications after the fact. There is little middle ground. However, one useful method which is often overlooked in development methodologies is the Operational Readiness Review (ORR). This approach calls for stringent solution reviews prior to launch. The goal is to achieve both a smooth launch, a low defect rate, and improved availability in the operational environment. Many IT Operations teams in the industry employ an ORR process but this can be expanded [1].

This paper discusses the background of the ORR, its origins, recent developments in Site Reliability Engineering practices, and current approaches. In addition, multiple generations of the development and deployment of actual ORR processes are discussed. These experiences became the predecessors for a newly created Global ORR process in a major commercial IT organization documented here. The development of this Global ORR process, its universal adoption, management dashboards, and reduced operational error rates aided in the realization of smoother post-production experiences. Of particular interest is the fact that this implementation represents a truly Global approach both geographically and organizationally. Additionally, this process implemented an automated data collection and Global dashboarding process.

II. ORR DESCRIBED

The ORR approach has a long history and has been known by different names. The development of the specific Global ORR process being presented here emerged from earlier verification processes [2][3]. These precursor approaches applied an architecture review, a non-binding readiness review early in the lifecycle, and a binding final readiness review. The PRE (Production Readiness Exploration) and the PRR (Production Readiness Review) were used on dozens of application releases and produced strong quality protection and improvement results.

Based on the success of these processes a new version of the approach was recently developed which was called the ORR (Operations Readiness Review) as this is the standard ITIL nomenclature [4]. The definition of the ORR can be summarized as:

“…a process of preparing the custodians of an asset under construction, and their supporting organization, such that, at the point of delivery/handover, they are fully ready to assume ownership of the asset.” [5]

The process employs a checklist (partly automated) which is used during both development and service integration/deployment preparation to prepare technologies for usage. If a review does not pass, applications or deployments require further mitigation and adjustment. Furthermore, the possible failure scenarios may expand or change. Where there are limited processes and procedures supporting the development and achievement of Operational Readiness, unprepared platforms and environments are the result. This can lead to operational gaps, system faults, and costly corrections. Thus, the ORR is a vital step in the service realization process.

III. ORR PROCESS BACKGROUND

The Operational Readiness Review’s roots stem from standard quality processes and signoff checklist procedures often seen in manufacturing and other domains. Variants of the ORR are found in disparate industries beyond IT including chemical manufacturing [6] and nuclear power safety processes [7]. ITIL (IT Infrastructure Library) defines the ORR as part of the Service Acceptance Criteria (SAC) [8]. Within this process the ORR helps ensure IT services meet quality requirements and that the service provider is ready to
operate a newly deployed service [9]. The essential form of the ORR is shown in Fig. 1.

Within software and systems development, the ORR has evolved into a standard process practice and is formalized within ITIL. According to Yakimoff [10]:

“The key objective is to ensure the service is working towards readiness for operations to assume full ownership. This activity also helps to provide assurance to stakeholders, and there is sign off and acceptance from the Operational team. It can also identify and manage any risk during the review process”.

Typical ORR methods aim to prepare systems for release, reduce risk, and inform stakeholders of preparedness. Reviews can be narrow or broad, yet they often will include coverage of designs, testing, computing environment, operational plans, support, maintenance, and DR (Disaster Recovery) [10]. The ORR is also promoted by Project Management bodies like the PMI (Project Management Institute). The PMI takes the position that it is the project manager’s responsibility to ensure a readiness program and perform required assessments. These actions are considered as necessary deliverables for any project [11].

IV. CLOUD PROVIDERS AND ORR PROCESSES

With the increase in dominance of several major Cloud hosting providers in recent years most of them have also adopted a form of the ORR. For some companies they follow a standard ITIL model while others have customized their approach to suit their operational needs. In the case of Microsoft Azure’s approach to the ORR their readiness process focuses on Security, Monitoring, High Availability, and Disaster Recovery [12].

For Amazon Web Services (AWS), a significant emphasis on the ORR to confirm applications are ready for production operations through their “Operational Excellence pillar” of their AWS Well-Architected Framework process is promoted [13]. The AWS ORR includes reviews of testing approach, monitoring, and an approach for measuring performance against the application’s SLAs. Interestingly, AWS also ensures that applications can report data on service interruptions in a nod towards resiliency [14].

At Google, IT Operations are more commonly referred to as Site Reliability Engineering (SRE). Their pioneering work in largescale Cloud computing and operations has made them a trendsetter in this area. In a landmark book on SRE, Google lays out their computing environment management and engineering methods in detail [15]. The launch preparation method described in this book resembles standard ORR methodologies. Google’s version of a PRR process itself appears in the SRE documentation. In addition, Google’s approach plays a key role in their DevOps methods.

Finally, Hornsby [16] adds several key considerations around the ORR for Cloud architects. He mentions that the ORR is rigorous, and evidence based. It will also be specific to a company’s environment, culture, and platforms. Regardless of this, the ORR is always meant to uncover gaps and reduce risk to operations. This is why a broad set of contributors and stakeholders ought to be involved. In the implementation of the Global ORR process described below, the foundational inclusion of the PRR and ORR by the major Cloud operations providers and their DevOps processes provided a key influence.

V. DEVELOPING A GLOBAL ORR PROCESS

The effort outlined here was undertaken to develop and introduce a Global ORR process at a major information services company to improve system reliability and availability and to prevent unexpected environment anomalies. This environment was truly global in that it operated in dozens of countries and serviced customers throughout the world. The scope included a portfolio of over 3,000 applications running on tens of thousands of Cloud-based servers [17]. Heterogenous platforms and significant legacy, new development, migration to cloud, application onboarding of merged/acquired companies, further characterized the complexity of the environment. The Global IT process engineering team was tasked to develop the ORR approach at the request of the IT Operations Division CTO and CEO. IT Operations teams across multiple Divisions around the world provided input to the ORR development and deployed it over time to establish it as a new standard practice. This process development method is reflected in Figure 2.

A. Process Goals

As stated, the intention of the new Global ORR process was to enhance operational quality and system Availability to prevent service interruptions on behalf of customers. The specific goals informing the program included: 1) drive uniform systems quality level with explicitly defined criteria; 2) obtain Operational and Availability objectives and infuse continuous improvement throughout the systems operations process; and 3) provide transparency around support requirements and strengthen sharing of best practices across Global teams.

In addition to these primary drivers, it quickly became clear that the process would need to address the nature of the Agile and DevOps lifecycle models in use throughout the company. The implementation of the ORR process had to balance a comprehensive checklist with a DevOps lifecycle which sees itself as highly Agile. This was one motivation for closely reviewing Cloud based operational SRE processes which incorporate a PRR as standard practice while still retaining its DevOps approach.

B. ORR Process Development Steps

After defining the need for a Global ORR process and setting in place the leadership support and objectives, the development process was planned and initiated. The approach
to developing the process is summarized in Fig. 2. The major steps of this development included the following:

1. **Initial Checklist Creation**: Some Divisions did not have any pre-release checklist process. However, there were three internal departments with preexisting operational checklist procedures in place. These were merged to seed the new Global ORR. Best-in-class methods were harvested and started the buy-in process as those groups with operational checklists felt included in the development of the new Global process.

2. **Industry Research**: The team scanned available literature for current practices. Starting with the ITIL framework, carefully reviewing Google’s SRE methods, and searching for other examples of ORR approaches all provided additional input to the definition of the Global ORR approach and checklist.

3. **Executive Reviews**: As the process and checklist emerged it was reviewed with senior management. Their support for the process was unequivocal and they provided pointed feedback on improving the approach and their expectations on its usage.

4. **Pilot Phase**: The deployment involved a small group of willing participants who had prior experience with PRRs or ORR-like preproduction checklist processes. This audience provided a testbed for the process as well as critical ideas on improving it and the checklist itself. These learnings helped tune the process artifacts and prepare for a broader rollout.

5. **Self-Reporting Phase**: Following the Pilot Phase all application teams in the initial population were required to participate in a “self-reporting” period. This allowed teams to begin picking up the process without tight oversight or data auditing.

6. **Full Deployment**: As the initial process was stabilized, a phased approach was followed to ensure adoption from all Divisions. With the support of leadership, a governance approach was defined and followed. Additionally, project release readiness was reviewed by management as a condition of production release.

7. **Standard Global Practice**: Over the course of approximately 18 months, the process proceeded from concept to design to pilot to a required Global Practice adopted by all divisions. The checklist itself entered planned and versioned releases to introduce improvements uniformly and globally without creating disruption in ORR usage. Also, governance migrated to the Global IT PMO (Project Management Organization) group in coordination with the Architecture review team.

### VI. ORR METHODOLOGY AND TOOLS

In terms of the specific features or characteristics of this Global ORR process and checklist there are a few notable items listed below beginning with the composition of the checklist itself. The checklist evolved into a proven list of just over 100 quality checkpoints. The initial categories included: Capacity, Performance, Touchpoints, Batch, Backup/Recovery, Support, Network/Firewalls, InfoSec, Cloud Computing, Monitoring, and DR.

During the 6-month roll-out of the checklist, the categories were re-grouped based on the department/function to simplify follow-up and verification processes by the respective function. Revised categories emerged as follows: General, Application, Hosting/Cloud Computing, Network, Database, InfoSec, and DR. Each section contains specific elements related to capacity, performance, support, monitoring tools, and additional details.

#### A. Checklist and Checkpoint Details

The number of questions (or checkpoints) in these categories range between 59 and 67 depending on the change type. A summarized listing of the checkpoints used for determining release readiness are shown in Table 1. This listing is a subset of the application-oriented checklist. Proprietary information and tool names have been removed and the wording is simplified for space reasons. However, the essence of the checklist is represented and can be expanded upon and applied to build customized checklists using localized checkpoints relevant in any organization.

For this approach, release authorization required a score of 100% against the checklist. The questionnaire was largely automated and included a branching feature to reduce work depending upon the architecture/environment. The upload, data extraction, and visualization were fully automated.

The ORR scoring included a color-coding scheme as follows:

- **Green** (approved): 100%.
- **Amber** (conditionally approved): 70% to <100%.
- **Red** (rejected/requires rework): <70%.

Any score below 100% required an approval from the Executive stakeholder team to proceed with the release. In such cases, the Division Head assumes responsibility for the risk of not meeting the quality criteria. Importantly, any passing score must achieve 100% fulfillment of the checklist. Any lower score would fall into the Amber or Red category (see Fig. 3). To compute the score (1) is applied:

$$ R = \left( \frac{\sum_{i=1}^{n} (C_i - O_i)}{T_i - O_i} \right) \times 100 \quad (1) $$

where

$$ R = \text{ORR Score} $$
C = Affirmative checkpoint responses  
O = Out of scope checkpoints  
T = Total number of checkpoints

The required checkpoint characteristics include:

1. A reference number.  
2. Checkpoint description.  
3. Readiness Status with sign-off.  
4. Name of accountable party.  
5. Verification support.  
6. Resolution Date.  
7. Remarks.

The scope of applicability for the ORR covers all major releases or changes and a required process for major application implementations, onboarding new systems, hosting transitions, or other significant changes. The process was defined as optional for hotfixes or minor enhancements with low risk. Specifically, the ORR applicability is outlined here including which conditions call for mandatory usage:

- New application implementation.  
- Major application releases.  
- Onboarding of acquired applications to the central environment.  
- Hosting Model change.  
- Hosting Supplier change.  
- Infrastructure replacement change (e.g., OS upgrades, Database upgrades, storage platform changes).  
- Multiple application releases must maintain an ORR per application.  
- Application patching release can be granted a waiver depending on assessed risk.

B. ORR Checklist Dashboard and Usage

Executive Stakeholders requested the ORR Dashboard to see an at-a-glance readiness view. The design metaphor used for the dashboard was that of the “Christmas Tree” panel on a submarine [18]. Prior to diving under the water, this mechanism shows all green lights if hatches are sealed. If there is a red light, then something needs attention. Similarly, the ORR dashboard automatically summarizes quality checkpoints into a simplified graphic with both quantitative values in terms of percent ready as well as color coded status (red, yellow, green) based on defined thresholds (Fig. 4).

The ORR release preparation approach does not eliminate the Change Management process but strengthened it and reduced overall risk. Within the ORR process there was a defined Change Owner, supporting teams, Change Manager, and Authorizer. Moreover, getting to full adoption of the process took time. Progress proceeded group by group and system type by system type and eventually reached universality. In addition to automated branching, project teams upload the checklist to a central repository where automatic parsing of the checklists using PowerBI enabled automatic generation of the dashboard.

VII. DEPLOYMENT CHALLENGES

Realizing a process change of this type across a large global organization requires a project approach running through phases, including building the right climate for change, enabling the teams to succeed with the change, and finally, implementing and sustaining the change [19]. During the process design and deployment several key concerns were heard (some of them frequently) from the teams and organizations adopting the ORR process:

- The first complaint was that the ORR process added unnecessary overhead. Certainly, the ORR represented a new quantum of work. Yet, the upfront effort level was known. Once teams began using the ORR, they did see the benefits as they observed hidden preparation gaps and prevented major issues in the field. The execution time of ORR was initially high due to team’s unfamiliarity with the checkpoints, and additional time was taken in correcting the hidden preparation gaps. Soon, teams realized the benefit of using ORR which prevented major issues in production. Once teams became familiar with the checkpoints, they started integrating most of the checkpoints in the design phase itself. This effectively reduced the overall development time taken thereby reducing the cost of correction towards the end and paid off in less firefighting post-production.

| Table 1. Sample Checkpoint Listing |
|------------------------------------|
| **General**                        |
| 1. Requirements signed-off?        |
| 2. Architecture signed-off and applied to enterprise repository? |
| 3. New and modified code reviewed and signed off? |
| 4. All transition documents completed and support informed? |
| **Application**                    |
| 1. Application portfolio repository updated? |
| 2. Third party software updates completed? |
| 3. Shared code or configuration settings communicated? |
| 4. Automated build for application available? |
| 5. Software licenses and support available for all components? |
| 6. Required batch run time schedules reviewed for conflicts? |
| 7. Are all applications added to standard monitoring tools? |
| 8. Has performance testing met threshold requirements? |
| 9. Has performance of all SQL been analyzed? |
| **Hosting/Cloud Computing**        |
| 1. Has Load-Capacity Planning been done? |
| 2. Are Load Tests metrics within the expected limits? |
| 3. Are all required backups and notifications running? |
| 4. Are servers patched to latest levels and in patching schedule? |
| 5. Has OS Hardening procedures been completed? |
| **Network**                        |
| 1. Has Firewall impact analysis been conducted? |
| 2. Has Load balancing been tested and confirmed? |
| 3. Has cluster load balancing been reviewed, optimized, and tested? |
| 4. All servers verified for connectivity to all DNS servers? |
| 5. Are IP addresses changes applied to all affected components? |
| **Database**                       |
| 1. Have database changes been reviewed and signed off? |
| 2. This includes, instances, DDL changes, DB code, SQL, etc. |
| 3. Are DB names, schema owners, object owners identified? |
| 4. Has space utilization been reviewed, and actions taken? |
| 5. Is DB monitoring for events failures with alerts defined? |
| **Infosec**                        |
| 1. Are security all servers installed with latest Antivirus? |
| 2. Do all secure connections use latest protocols? |
| 3. Internal and external Vulnerability Scans performed? |
| 4. Have SAST scans been performed where required? |
| 5. Have Open-Source Scans been performed? |
| 6. Have any Security findings been solved? |
| 7. Has environment been onboarded to SIEM monitoring? |
| **DR & Failover**                  |
| 1. Has the environment plan been reviewed for DR readiness? |
| 2. Has Failover testing been performed and verified per plan? |
• Another viewpoint from some project teams was that the ORR appeared more "Waterfall" than "Agile". To resolve this the process team leaned on the integration of DevOps process considerations while authoring the checklist and the process of conducting the ORR. The fact that the leading Cloud providers used ORRs was a powerful example as well.

• Dependency on Operational Support teams for verification posed some difficulties. Some checkpoints could not be exclusively handled/verified by the Application teams and required assistance from the Operational Support teams. There were often no defined communication channels to contact the responsible person. To address this situation, key Point of Contact (POC) and communication channels were identified to verify the readiness of the particular checkpoint. This further improved collaboration between the Applications and Operations teams (in essence, a core tenant of DevOps).

• Some teams claimed platform specific exemptions. Such views of non-applicability came from ERP systems teams especially. In response, a tailored template incorporating relevant platform points was developed. The dynamic branching feature of the checklist questionnaire also reduced this issue.

VIII. ADOPTION ENABLERS
In addition to making adjustments to the challenges noted above, there were some factors which provided natural support for the roll-out and adoption of the process.

• Positive Influencers: In general, few project teams immediately saw the benefits of the ORR process. The fact that the process was mandated by senior management did not eliminate pushback yet it did help to drive up adoption.

• Release of Frequently Asked Questions: A common set of questions centered around when to use the ORR and who was responsible for completing the checklist. Many checkpoints required Verification support from other teams/departments. Since the application of the ORR was essentially universal the first question was easy to dispatch. To manage the second question around responsibilities a complete RACI matrix was developed and reviewed with all stakeholders and project teams.

• Evolution of a stable ORR process: Finally, as the ongoing development and evolution of the ORR process and checklist was a bi-directional community effort, buy-in was enhanced. Leaders/Subject Matter Experts and teams found that they had a say in what the checklist contained and how to apply it. Dedicated efforts were made to incorporate suggestions from the detractors to improve efficiency of the process. This broke down resistance to the process as ownership of the approach became democratized throughout the organization and results began pointing to improved system preparedness and fewer production outages.

• Continual Improvement: SMEs (Subject Matter Experts) were given the ownership of continual improvement of the checklist for their sections. With the ownership in their hands, quality criteria could be refined based on the evolving nature of technology and lessons learned from the operational issues. For example, Hosting and Network checkpoints were owned by Hosting SMEs and Network SMEs respectively. The checklist updates are owned by the Process team based on the inputs from SMEs. Monthly Governance meetings with these SMEs helps to identify the enhancements to the checklist.

IX. RESULTS AND DISCUSSION

Upon reaching the steady state phase the Global ORR process was used throughout the organization and on releases or changes as specified in the process criteria. Since the portfolio of applications reached into the thousands, and most applications released often due to their Agile SDLC approach, this resulted in hundreds of ORR reviews per month around the world.

To support this approach dedicated process engineers worked part-time to take in feedback from users and developed improved checklists and expanded process automation. Additionally, a Project Manager provided overall governance support and members of the Architecture group provided technical review guidance. And of course, the project teams themselves prepared the checklists, submitted them for review, and achieved approval sometimes incrementally. Thus, running the ORR process is not without an investment in staff hours.

However, the rewards of the process became apparent quickly. Going back to the original goals of the process, the ORRs did push a common set of technical standards into all projects uniformly. Also, the company witnessed an improvement of production release readiness and a decline in post-production environment related issues. Moreover, the ORR process had an impact on the engineering culture of the organization. A top to bottom awareness of what operational system requirements were, became more commonplace and teams began thinking about the questions on the ORR early in the lifecycle (much like the previously mentioned PRE had recommended). While the Operational Readiness Review (ORR) is commonly viewed as a quality assurance measure prior to release, its adoption actually facilitates the integration of quality into the design process itself. As teams become accustomed to these expectations, they naturally develop a mind-set that embeds quality much earlier in the lifecycle which is essential in reducing costly error corrections towards the end.

In fact, the term “ORR” became part of the lexicon of the IT organization. This “left shift” of quality meant that design time thinking was applied to questions of performance, security, reliability, and support in advance of the actual ORR read through. This then sped up the process even further than the automation did.

Perhaps one of the most successful parts of the Global ORR process was its ability to inform senior management of the status of each in-flight release and provide a quantitative and succinct visual view of both preparedness and risk levels. Prior to the Global ORR each project team or Division followed their own process for release preparation. With the ORR there was now a common language and data representation across all applications making it easier for management to assess risk, provide support, and communicate with various stakeholders regarding status. The visualization dashboard condensed a wide array of technical information...
into a concise decision support tool as in Fig. 4. In this approach rapid understanding was achieved and approvals or adjustments directed by senior management. The practical application of this process and governing dashboard have been used on hundreds of production releases globally.

A final point which is worth emphasizing is that the ORR does not replace existing Change Management practices. It is instead an additional risk mitigation and quality control mechanism. ORR practice can be effectively integrated with Change Management as the final stage gate (Go/No-Go) before deploying into Production. Effective design methodologies and testing techniques are still required to achieve High Availability. However, the ORR can and does drive key technical and operational considerations and requirements into the application and system delivery process. Such an approach also requires a high degree of sophistication by individuals in the organization. A wide range of diverse topics need to be covered in the review process and a certain maturity is required to adopt this type of process which calls on people to step back and constructively critique their own work and that of others to improve the ultimate solution.

X. CONCLUSIONS

The story of the ORR in general and the development and use of this instance of a Global ORR is a powerful one in the reliability toolkit. It can prepare solutions far in advance of deployment for the appropriate environment setup and it can prevent costly errors from entering production. This method does require commitment and investment to get started and to maintain but it also has significant payback including the capture of tribal knowledge regarding the important questions to ask about an engineered solution, platform, environment, or problem domain. This process is a true balancing act between expert knowledge, process, and technology.

Existing industry checklists and the Global ORR described here are very similar. What is critical is that within a DevOps environment like Google, a rigorous PRR or ORR method is expected. As an industry standard method, the ORR approach can be seen as a supporting model for achieving high levels of operational quality, reliability, and availability. Such a process can also improve Cybersecurity compliance, application performance attainment, and more. This paper has outlined the initiation, development, deployment, and use of a practical ORR model and tooling environment utilized on hundreds of production project releases. Key success factors include Executive support and visibility to the ORR data, an active Change process to support adoption within the culture, and a universal application of the process to achieve even higher levels of systems quality and availability.