INTEGRATED DISSEMINATION PROGRAM – TECHNICAL APPLICATION STANDARDS

Version 1.6.0

This document explains the technical requirements for applications running on the IDP.
The Development Organization must conform its applications to these standards.

TABLE OF CONTENTS

Revision History

Version	Date	Author(s)	Revision
0.5	9/2018	A. Al-Mallah	Created
1.0	7/30/2019	C. Klemmer & S Earle	Document Released
1.1	9/6/2019	OBT	Minor updates, including logging and environments
1.1	1/8/2020	OBT	Included version specific software, embedding code practices, and handling site differences
1.1	1/8/2020	C Klemmer	Released
1.2	8/5/2020	C Klemmer	Updated to include database testing, user sign on, languages, update WWA failover, load testing expansion
1.3	3/8/2021	J Huber	Updated Monitoring section, added general coding standards to software design section, Added end-to-end testing under Testing and Validation, clarified supported versions of databases. Added LDM verbosity requirement
1.3.1	5/19/2021	J Huber	Added performance metric requirement to testing and validation section
1.4.0	9/15/2021	J Huber	Add delivery instructions for development organizations, consolidated installation pieces into its own section, re-wrote testing section, moved and renamed DataFlow to Data Standards. Updated logging section to provide guidance on how systemd logging should be handled. Improved application error handling section.
1.4.1	12/1/2021	J Huber	Added templates for documents required for delivery.
1.5.0	3/30/2022	J Huber and C Klemmer	Add web analytics and cloud standards
1.6.0	5/18/2022	J Huber and C Klemmer	Reformatted the flow of the diagram. Updated to indicate RHEL8 is our standard. Rewrite to make the document more adaptable to NCO standards for non-IDP programs. Added standard for data sync to AWS

This document describes the technical requirements for the Development Organization (DevOrg) to follow when designing, upgrading and optimizing the applications that run in NCO that include the Integrated Dissemination Program (IDP) and the Compute Farm. Prior to any system builds or code handoff to NCO this document should be reviewed and all necessary changes should be applied. If for any reason you believe you cannot comply with anything within this document please speak to the On Boarding Team (OBT) Lead. They will work with you to determine a path forward into operations.

Hardware Design

Cluster Configurations

There are 2 defined clusters, one will be dedicated for applications that directly support dissemination of Watches, Warnings, and Advisories (WWA) and the second for everything else unrelated to WWAs. The only applications that can be moved into the WWA cluster are those that are directly responsible for the processing of WWAs or their entire purpose is for the dissemination of WWAs. Anything else must be approved by the Director or Deputy Director (DoDD).

Hardware Specifications

The specifications for each Virtual Machine (VM) build will be provided by the DevOrg or customer and must contain:

- VM name that follows the [IWSB Naming Convention](#)
- Required number of Virtual Central Processing Units (CPU)s
- Required Virtual Memory
- Required RAM disk
- I/O traffic footprint (either an estimate or raw numbers if a port from another location)
- User and application accounts
- Public or Internal IP requirements

Both CPU and memory requirements should be built with an expected percentage utilized at 80% under maximum load or maximum weather driven event. The recommended number of CPUs should not exceed 4 vCPU for a non-database VM and should not exceed 6 vCPU for a database VM.

Operating Systems

The following operational operating system is supported on the IDP infrastructure:
- Red Hat Enterprise Linux: Version 8

Disk Specifications

The following specifications must be provided for each disk an application accesses:
- Requirement for any local disk space
- For shared NFS mounts and local mounts the following information must be specified in the software requirements:
  - X Name that will reside under:
    /common/data/apps/*application_name*/*tier*/*X*
    Where *application_name* is how you are identified on IDP
    Where *tier* is either "dev", "qa", or "ops"
  - Size
  - Maximum number of files (inodes)
  - Latency requirements
  - Peak I/O generated by application

Web Services

The following is a list of the web servers that are supported on the IDP infrastructure:
- Apache: Version 2.4
- TomCat: Version 7

IDP servers data via HTTPS or FTP front ends. All IDP web applications reside in a shared front end infrastructure of load balanced web servers. If the application requires a dedicated front end infrastructure, appropriate justification needs to be provided to the OBT lead.

All websites will be hosted out of Akamai with the College Park and Boulder data centers serving as the origin locations.

Website Analytics

Per OMB M-17-06 all public federal web sites are required to use the federated analytics functionality. In practice, only websites that have a presentation layer can exercise the analytic capabilities. Applications that provide only raw directory listings or return only raw data will have the usage tracked in accordance with the web-based logging retention standards below. Additionally, for public web applications that utilize apache (httpd service), the logs are required to be set up for review in awstats which is maintained by IWSB.

(NOTE:
https://www.whitehouse.gov/wp-content/uploads/legacy_drupal_files/omb/memoranda/2
017/m-17-06.pdf
And https://digital.gov/guides/dap/add-your-site-dap/)

Data Requests

Applications to be onboarded will likely have input data requirements as well as output
generated by the application. This section focuses on what data is required, how new
data can be retrieved, and how application generated output can be made available.

If the data is valuable on an enterprise-wide level then the IDSB Dataflow Team will be
responsible for all data movement. If the data is only valuable to the application alone,
then the application itself will be responsible for the data movement.

Input

All operational applications shall receive inputs from operational systems and never
from QA or development systems. If non-operational data is required from an external
partner it must not be considered critical within the application usage.

All input datasets used by the application must be described, including:
- Type of data (*model, observation, etc*)
- Source (*NOAA, other government agencies, commercial, etc*)
- Source Type (*FTP, HTTP, LDM, etc*)
- Update Frequency (*hourly, daily, etc*)
- Impact of data being unavailable (*output degradation, application unable to run, etc*)
- Any restrictions to data (*sensitive data from commercial or foreign government source*)
- Any agreements relevant to the retrieval and use of the data.
- Level of support for data inputs (24x7, 8x5, etc.) and points of contact.
-

Requests for data must be submitted to the Dataflow Team using this request form:
https://docs.google.com/forms/d/e/1FAIpQLSevecbKGt-T3uTSdRdvsvw-KqtDKadBKH1
UG_2UHZIM_nRJtA/viewform

Output

All output datasets created by the application must be described, including:
- Type and Format of data (*Radar GRIB2, Satellite NetCDF, etc*)
- Individual and aggregate size of data (*ex: Between 1 - 10MB each, keeping 1000 files, disk usage up to 10 GB*)
- Dissemination platforms (*FTP Server, LDM, http service, etc*)
- Requirement for amount of data retention. If the application requires an archive, there needs to be a resource commitment to buy disk for the archive.
- Restrictions on distribution and use of the output data.

User Data Access

All external users who need to authenticate/authorize into the application should use the enterprise solution - RedHat SSO. This software package is in beta right now, please reach out to OBT if this appears to be a solution that satisfies any application requirements.

Cloud Usage

If there is a request for NCO to support an application in the cloud it should first be coordinated with the Office of Dissemination with a follow-on conversation with the DoDD and IDSB Branch Chief.

If there is a request for IDSB to transfer data to the cloud it must follow the guidelines below:
- Any request to add, remove, or modify data syncs to the cloud must be first approved with the Office of Dissemination prior to engaging IDSB.
- The only cloud provider that NCO is currently working with is Amazon Web Services (AWS).
- Any data request must be for data that already exists on IDP.
- If an existing data partner is requesting a feed to the cloud it must:
    - Use any current data exchange software methods used today
    - Be a one to one switch for the current exchange

Technical Requirements

The Onboarding and Dataflow teams maintain a data sync of requested products on IDP. At this time the data synced is upon request only.

- The syncing is only done from one "active" side at a time using the dedicated sync servers.
- Only one copy of the data will be provided to one drop point. Users are responsible for fanning it out from where the data is synced to.
- The frequency of data syncs will be no faster than every five (5) minutes.
- Monitoring of the sync is based on the far-end services level. For example, if the far end data is only maintained during normal business hours, NCO OMB and NCO OBT will only provide support during normal business hours.
- Data requests should include the following:
    - Source of the data (IDP application output, individual mount point, etc.)
    - A regex or some other identifying information on what data should be pushed.
    - The expected average amount of data that will be pushed in a 24 hour period
        - This should include the peak amount of data to be transferred at any one point during this 24 hour period
    - The peak amount of data expected to be pushed in a 24 hour period, if different from the average.
    - The frequency of which the data is produced.
    - The cloud credentials (shared via Kiteworks) and storage destination.
        - This includes the username, password, bucket name, and path
    - Point of contact for far end delivery point.
    - If the request is being made by an application owner already on IDP, an updated roles and responsibilities document must be signed to add cloud data support.
- All new requests will be monitored for a period of ten (10) days in the OBT QA syncing environment. After successful completion of this period, a Request for Change (RFC) will be filed to add the data syncs to the operational sync process.

Application Monitoring

The standard monitoring for all applications, hardware, and databases will be done within the Big Brother tool managed by NCO. Our monitoring is driven by three categories, "green," all is good, "yellow," a warning that is acted upon with 8/5 support,

and "red," which is supported urgently with 24/7 support. For example, the monitoring is triggered on the support level end-to-end. If an incoming data set is only supported 8/5 by the remote end, the monitoring on the NCO side will match that support to only by 8/5.

 NCO is responsible for configuring all scripts based on the requirements provided by the DevOrg or customer:

- All of the identified critical components of the application. The monitoring should not include both the processing and the output. Only one component of each application will be monitored. Examples of what to provide include:
  - When the data or processing is late
  - What steps to take or who to contact if something flags red.
- List of critical processes that comprise the application that should be monitored
  - Include if the processes runs all the time, or if it has a cadence of starting/stopping
  - A few sentences of documentation notes of what each process does
- List of web sites or other external points of access (sites required to be available to the public)
- Log location on disk
  - Error message pattern to search for in the log
- Nominal disk usage for local files, excluding shared data files

NCO Operation and Monitoring Branch (OMB) monitor's Big Brother systems 24/7/365. In the event of a problem, the 24x7 staff will have contact information for the on-call personnel in the Big Brother information pages. The on-call personnel will troubleshoot the issue and will, if necessary, contact the DevOrg for assistance.

Software Design

System Components
Do not embed any code that should be in an application centralized repository. Doing so prevents security patching from being executed. This includes any software that is downloaded through the official RedHat Network Channel. Note that when it comes to designing any recurring process a check shall be made to ensure that an instance of itself is not already running before it kicks off.

Site differences
It is required to keep system and application configurations the same between College Park and Boulder sites. If they need to be different (eg:due to IPs or VM names, etc),

these differences have to be documented in the top of the configuration file and easy to find. Do not embed differences within the configurations. Where possible, keep the scripts identical between sites and use a reference pointer to a centralized configuration file.

General coding

This section applies to all scripts and compiled code used by the application

- No code will have hard coded paths. All paths to get to input or output data will be read from environment variables or configuration files
- Variables used to define an environment (ops|qa|dev) will be read from config files
- Any site specific definitions also will be read from config files

Scripting

This section applies to those scripts used by the application or used to maintain the environment

- The first line of the script will always identify the SHELL being used
- Every script must contain a clearly demarcated comment section that explains what the script does, what its expected inputs are, what the outputs are and what the errors/exceptions thrown are.
- All scripts (bash, perl, python) permissions will be set as 775
- Any script utilizing command line utilities as-is must use BASH
- Scripts must always return a predefined status value
- Errors/Exceptions are echoed to stderr and written to log files
- Any function that is not both obvious and short must be commented
- Use blank lines between blocks to improve readability. Indentation is two spaces, don't use tabs.

Languages

The application shall be implemented in any of the supported tools/languages/frameworks:

Compiled Code	C++, Java
Web Application	Java, PHP, Perl, JavaScript, HTML5
Interpreted Code	Bash, Perl, Python (Anaconda)

- If any application were to use an enhanced framework in addition to one of the above listed languages, an appropriate justification needs to be specified. Support agreements for such frameworks need to be in place before they can be used in IDP and discussed with the OBT lead.
- These external frameworks may need further review by the NCO security team and the OBT. The most likely questions regarding external frameworks are their support structure, and security aspects. Another consideration would be to evaluate open source equivalents, if available

Shared Software

If applications are leveraging shared software between them it needs to be kept in a common NFS area where both systems can point to it. It is not to be installed locally. A minimum of 2 versions of the software will be kept at all times. Applications will migrate to the latest version with a planned upgrade.

Queuing

Use standardized mechanisms for common applications such as queueing and clustering, as opposed to using custom applications to achieve the same result. The best practices shall be adhered to where available. The following queuing mechanisms are allowed on the IDP infrastructure:

- Local Data Manager (LDM) Version 6.13
- RabbitMQ (https://www.rabbitmq.com/)

Databases

All databases need to be able to drop all tables and start from scratch at any time, without any negative impacts to the functionality or stability of the application.

The following database servers are supported:
Relational

- Postgres Version 9.0 or higher
- Mysql Version 5.6.x

Non-Relational

- MongoDB Version 3.4

For new system builds, the below versions are supported:
Relational

- Postgres Version 14.x or higher
- Mysql Version 5.7 or higher6

Non-Relational
- MongoDB Version 4.4 or higher

Compilers/IDEs

The following compilers are allowed on the IDP infrastructure:
- GNU open source suite (g++) Version 4.8.5
- Java (javac) OpenJDK Version 1.8.0_x+
- Eclipse (IDE)

Build Tools

The following build tools are allowed on IDP applications:
- Makefile
- Jenkins
- Maven

Third Party Software

Before an application is onboarded:
- Developers must identify any 3rd party software used. Which version and the vendor support associated with it.
- Any 3rd party software used has to be approved by the security team
- Include any licensing costs which would have to be approved ahead of time by the NCO director. License funding will be provided by the devorg, or agreed to be covered by DIS.
- Open source tools must be considered, if available and a support agreement with a vendor established.
- Appropriate justification for the tool selected for a purpose must be provided. This could be a business case for using a specific software package
- All libraries should be from an authoritative source (ie Maven, pip, RH satellite server). If the software library is not available from such a source, the approved software form will need to be filled out and approved by NCO.
  - Specifically for Maven, the Open Web Application Security Project (OWASP) package must be instrumented in the build process. Further information can be obtained from the Onboarding Team.

Unsupported software

Software that is not allowed:
- Stand Alone Anaconda
- Oracle Java

Failovers

Applications must be designed to be able to failover to the non primary site within 15 minutes with no data loss and minimal data latency. An application that distributes Watches Warnings and Advisories (WWAs) must be able to failover in under 5 minutes with no data loss.

The failover must be designed to require minimal human intervention. The use of an automated "one button" script which switches the primary DNS to fail-over successfully is required. The application must be able to detect and report on whether the failover has worked properly within 10 minutes of the failover. An end to end test to confirm the products are being produced and received by the intended application(s) must be included.

Logging

Standardized application loggers are to be used and follow these guidelines:
- Application logs will be kept for seven (7) days. Any web-based logs will be kept for thirty (30) days.
  - Log retention periods needs to be configurable in a script
- Every independent piece of the application will have its own log file
- All logs, except current day's log, must be compressed using gzip
- The level of logging must be able to be changed in real time without recompiling the application. Instructions are to be provided on how to change the log verbosity.
- Logs will be written to the dedicated NetApp or local application volume; never under the local root volume.
- All application logs will have read/write permissions by the application user account and read permissions by group/world.
- Developers must use log rotate coding for a maximum of daily rotation
- All critical transactions will be included in the log output
  - Any "deletes," "writes" or "moves"
  - Any application that transmits data offsite must include a timestep of when the file was successfully sent
- Any application that transmits WWA's must have a more verbose output to fully describe the reason for the logged transaction.
- Any application that cannot use the system utility 'log rotate' must include a configuration file that determines where logs are stored, how often they rotate, and how long they are saved.
- LDM logs must be set to verbose by utilizing the -v option at start up.

- If systemd logging is being used, the application must use a dedicated log facility that is not the default. This allows for proper routing away from the default path.

Format of Log Cleanup
- BASH script will reside under $HOME/config called "log_cleanup"
  ```
  >retention_period=7 (defined as number of days)
  >find $HOME/logs -mtime -${retention_period} -type f -exec rm {} \;
  ```

Format of Log Contents
- Log filename must follow this pattern:
  app_function_YYYYMMDD.log
- Include date and time to the second for every log event
- Acceptable types of log messages: ERROR, WARNING, CRITICAL, DEBUG, INFO

Value	Severity	Keyword	Description	Condition
0	Emergency	emerg	System is unusable	A panic condition
1	Alert	alert	Action must be taken immediately	A condition that should be corrected immediately, such as a corrupted system database
2	Critical	crit	Critical conditions	Hard device errors
3	Error	err	Error conditions	
4	Warning	warn	Warning conditions	
5	Notice	notice	Normal but significant conditions	Conditions that are not errors, but that may require special handling
6	Informational	info	Informational messages	
7	Debug	debug	Debug level messages	Messages that contain information normally of use only when debugging a program

- Errors must be easily identifiable and machine readable in the log file
- Format example:
  YYYYMMDD hh:mm:ss, LEVEL, CODE_MODULE, MESSAGE

Application Error Handling

From time to time applications may encounter errors in normal processing. These conditions can be due to data integrity issues, application processing issues, or timeouts as a few examples. When this does occur the application should be able to complete its processing, write a useful error message to its log, and properly continue executing or stop and restart automatically.

Abusive Users

All software running on IDP must have all public facing components clearly identified so that parts of the application vulnerable to abusive users can be isolated and monitored. Taking action to block abusive users requires the coordination and collaboration of the OBT and IWSB and possibly NSB. The DevOrg shall design the application so that it is robust against abusive users and the internal actions of residing on a shared infrastructure.

Software Syntax Standards

This section applies to variables applications must use. Such as the home location for ldm, psql,etc.,:

- Keep variable names consistent when using capitals and underscores
  - Variables must be descriptive and contain one or more words
  - One character variable names must only be used in loops or for temporary variables
  - Prefix all names with "NCEP_" for example NCEP_LDM_HOME would have the home location for the LDM codelog
- Line length must not exceed 80 or 100 characters with no more than one statement per line
- Arrays must be constructed in a dynamic data structure, never static

Documentation

Templates

Standardized templates have been established and adopted by the OBT for all project artifacts. Key among those are templates for project Requirements, Requirements Traceability Verification Matrix (RTVM), Test Cases, and project Build Specifications. These templates shall be maintained in a standard location on Google Drive in folders for each functional area. During the kickoff meeting the applications specifications will be acquired and stored in the Build Spec Sheet Template. For clarification, please contact the OBT Lead.

Code Management

Application releases are to be version tagged based on a clearly defined methodology that is agreed between the Development teams and the IDP Onboarding team and must follow these standards:

https://www.nco.ncep.noaa.gov/idsb/Version_Numbering_Standard.pdf

Git is the mandated Supply Chain Management (SCM) platform. A VLAB/Redmine project must be created for every project that is onboarded, and appropriate Git repositories are to be tied with these projects

- A document that outlines how the application is laid out in the SCM system shall be provided by the Development Organization. Any changes made to the software code structure shall be accurately captured in updates to this document.

Code Installation

Installation Method
While not currently required, the preferred method for installation is to utilize Red Hat Package Management (RPM Package Manager) for all installations.

Installation Instructions
The developer must provide an Installation Document for each version release. This document must include at the very least the following:
- Step by step instructions on how to build the application. The build process must be automated using a script and instructions provided on how to run the script.
- Step by step instructions on how to install the application from scratch or as an upgrade to application depending on the type of release it is (new version vs. small upgrade).
    - An install script shall be provided
    - If the process calls for an install on several numerical systems, all systems must be listed explicitly. No shorthand in the instructions, IE. we don't allow: "vm-lnx-example1-40", a list of all VMs must be written out.
- Step by step instructions on how to back out the changes must be included.
- Instructions on how to verify a successful installation must be provided for every step
- Identify any environmental files that need to be sourced

Installation Scripts
The software Installation process must be config file driven. A script setting up the required environmental variables and what their value should be for the different configurations should be centralized to only be changed in one location. This will allow seamless installations between dev, qa, and ops.

Testing and Validation

Unit Testing

A standard practice of software development is to maintain a set of unit tests which target specific core functions. These tests are designed to ensure that any change to the software does not impact the core functionality. The expectation is that these tests are available for review and executed successfully before each software delivery.

Functional Testing

As a software release is nearing completion, the updated code must be tested with existing code base. This is done to ensure that all requirements are met for the changes being applied and that the updated functionality works in the system as a whole. These tests will form the basis of the test plan that is delivered for testing in QA and production environments.

Regression Testing

Regression testing is executed as the final checkout of the application. Regression testing is executed to ensure that previously fixed defects are not reintroduced by the updates being made. The development organization is responsible for creating, maintaining, and executing these test plans with each release. The results should be shared with the delivery. These will form the basis for the regression plans held and executed by the application support team for QA and production testing.

End to End testing

End to end testing with all upstream and downstream partners is a vital check to ensure that prior to installing in production, all partners have verified their ability to provide, receive, or process new data.

Development organizations need to keep track of and provide at delivery time any modifications to the following:
- specific partners, including contact information
- the data delivery or retrieval method

Onboarding will provide a testing platform to use which mimics production workflow.

Load Testing

When determining the maximum theoretical load, weather conditions at all times of year, and under all circumstances, should be considered as factors. For non-weather based

systems, such as emergency and telecommunications systems, a worst case scenario should be envisioned in determining maximum theoretical system load.

When code is delivered to the onboarding team the testing plan must include:
- Confirmation that you have done load testing in the Dev environment
- All representative endpoints to test load against
- Load test values for both standard days and critical weather days
    - This must be a maximum number of hits
- Acceptable response time thresholds for all endpoints

User Acceptance Testing

Developers must provide tests to confirm that the application meets all functional and performance requirements from a customer perspective. Each test should have the "action/input" to be done, and the "results expected", clearly outlined. Performance requirements include but are not limited to input data timeliness, data processing metrics, dissemination metrics, and any user based metrics (web service should produce x output in y time). These numbers should be backed by an analysis that is provided to NCO when code is delivered, if applicable.

System Testing

The OBT will perform appropriate scalability and trample tests no less than annually for each application. Trample testing is meant to expose weakness in both the application's ability to recover but also to improve monitoring of the system. Examples of this kind of testing include random shut down, loss of DNS, or read only file systems. Scalability testing coupled with a regular review of resource utilization are meant to balance efficient usage of finite resources with execution time of application level processes.

Failover Testing

Since there is only one site Each application must have prepared failover instructions maintained in collaboration with the OBT. If a failover procedure changes, the OBT will execute the failover during the QA period.

Security Vulnerability Tests

Prior to the submission of a RFC for operational implementation, the NCO Security team requires verification that the change being made does not have a negative impact on the security posture. A ServiceNow ticket should be opened with the Security team by using the Request Something New -> Security options. The "Web Scan" option should be used for applications that have a public facing website and "Network Scan" for applications that don't.

Software Delivery

Upon successful verification of the application code, the development organization will provide the following for it to be accepted for QA:

1. Software implementation plan that covers QA and production.
2. Release notes which cover the entire scope of the application upgrade.
3. Test plans that were successfully executed in the development environment. These test plans should cover all tests executed in the development environment.
4. All tickets opened by OBT will have been resolved or a waiver has been provided.
5. An email validating items 1-4 and an assertion that all applicable testing as laid out in the 'Test and Validation' section above has been completed successfully.
   a. If it is a major or minor release, we require an assertion email to say: All regression, unit, functional, load, and user acceptance has been completed successfully.
   b. If it is a patch release, we require an assertion email to say: All unit, functional, and user acceptance testing has been completed successfully.

Configuration Management

NCO follows very strict guidelines with all operational changes. Please review the configuration management documents for reference:
https://sites.google.com/a/noaa.gov/nws-ncep-nco-cm/?pli=1