

DAD IT POLICY





FOREWORD

Defence Accounts Department has been the pioneer in introducing automation and has made significant progress in computerization of its functions. Several applications have been built to facilitate automation and digitization. While these applications solved point problems but there is no uniformity in approach to IT software development. To take the digitization and automation journey forward in a structured manner, leveraging fast technology advancements and to drive standardization a need for an IT policy was felt. Accordingly, after detailed study of different standards in industries and MeITY guidelines on IT development, "DAD IT Policy" has been framed with the approval of Competent Authority. The DAD IT Policy formulates road map for IT development in the Department and aims to achieve standardization in software development and documentation. This policy is in supersession of earlier IT Policy.


(R P Singh)

Jt. C G D A (IT & S)

Contents

Background 3

Roadmap 4

Outsourcing..... 4

General Standards..... 4

Requirements Checklist: 7

Design Checklist: 7

Coding Checklist: 9

Testing Checklist: 10

Documents Checklist:..... 10

IT POLICY

Background

Over years CGDA has made significant progress in IT adoption, several applications have been built to facilitate automation and digitization. While these applications solved point problems but this has also led to data and application silos.

To take the digitization and automation journey forward in a structured manner, leveraging fast technology advancements and to drive standardization there is need for IT policy at the organizational level. Some of the benefits of such a policy are:

- Focus on standardisation: The standards ensure that the systems come with consistent quality and are comparable to the products or services in the IT industry. It will also helps in ensuring the security, interoperability, and compatibility of systems produced.
- Standards based project documentation: The documentation will be prepared by following the standards used in the IT industry. For example as recommended by IEEE, ISO and other commonly followed implementation methodologies. Documentations standards to ensure easy transfer of ownership of the systems.
- Real time MIS using modern technologies and integrated applications: The systems produced for the department should be integrated with other systems so as to produce real time MIS required at all levels.
- Single source of truth, data quality and availability: Multiple copies of data and its replications will be avoided unless absolutely necessary.
- Improved work distribution and collaboration: The policy focuses on improving the structures for work distribution between various divisions and improving the collaboration.
- Focus on improving core functions of organisation / domain and outsource non-core functions: Core functions of the department viz the payment, accounting, auditing and financial advice will be handled by the officials and non-core functions such as IT development and maintenance will be outsourced. This will bring in a specialised approach towards achieving stakeholder's satisfaction.
- Reduce cost by removing redundancies in IT infrastructure: The policy necessitates that the IT infrastructure be put to optimum utilisation at par with best practices in the industry.
- Improve user experience: IT systems will be continually improved to address the concerns of the internal and the external stakeholders (mainly the clients)

- Knowledge transfer and maintainability: In view of transfers of officials from IT to non-IT functions, it is important to ensure knowledge repositories, ease of access, availability and ease of transfer.
- Avoid dependence and lock-ins: Under this policy a Vendor lock in will be avoided as far as possible. This will be made possible by designing and adopting technology neutral solutions.

Roadmap

CGDA headquarters will establish a short term and long term IT roadmap, in consultation with field offices and stakeholders. Only projects which are approved in these plans will be taken up for development by organizations/team identified by HQrs. Small IT projects not in approved plans but of local significance which don't have wider organisational impact or use may be allowed after approval from the authority designated by CGDA.

Outsourcing

To be able to leverage technology advancements and remain focused towards domain functions, Outsourcing of IT projects will be the way of project execution and maintenance for IT projects. In house development by DAD officials will be allowed only in cases where compelling justifications are available, and only after approval of CGDA HQr office. By 31st march 2023 all In house development shall stop and maintenance of existing In house developed projects should be outsourced. This will enable CGDA to get skilled resources as per need and allow officers to focus on core functions.

General Standards

All IT projects, including maintenance projects, must follow certain standards as given below:

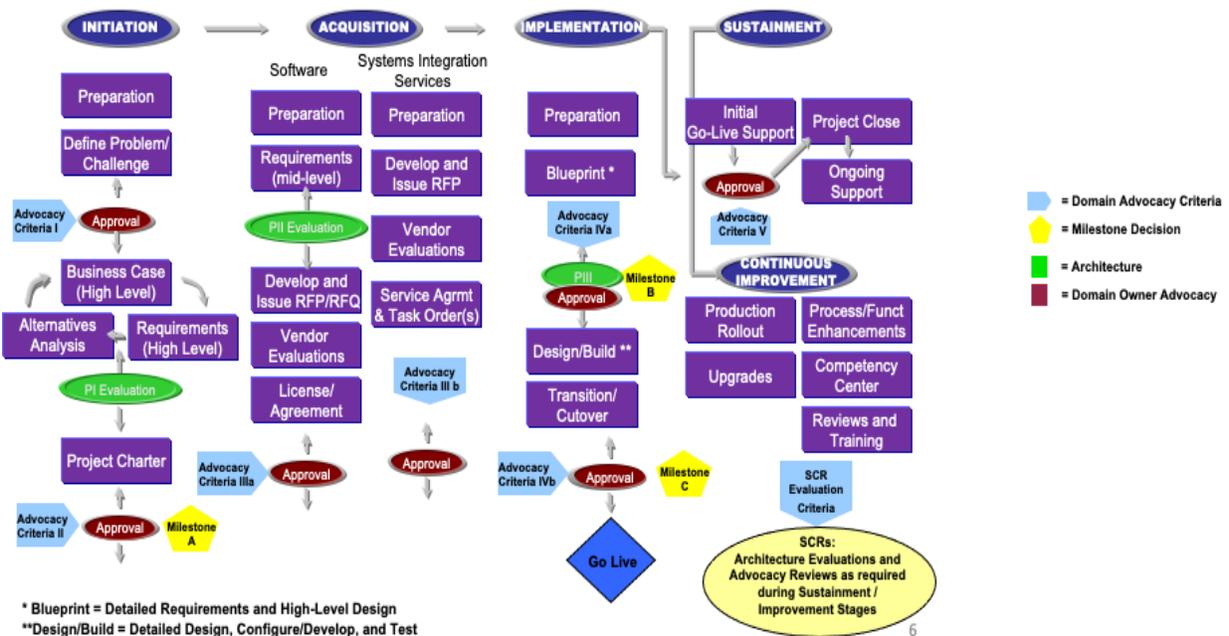
- Clear definition of purpose, problems the project will solve, objectives it is expected to meet, functions it is expected to deliver.
- Simplicity of use.
- Ruggedness (difficult to misuse, should not crash due to errors and give meaningful reasons for errors).
- Projects should be delivered on time to counter obsolescence and cost overrun.
- Reliability and consistency of the information produced by the IT system.
- Efficiency (fast enough for the purpose it was created).

- Conforming to current industry standards and Government of India and MOD (Ministry of Defence) guidelines issued from time to time.
- Clear, accurate and precise user documentation.
- Clear, accurate and precise technical documentation.

All systems must have designated project team who will act as Owners and Custodians of the IT systems. Project team will ensure that correct requirements are framed for new systems and for changes in the existing systems by involving all stakeholders. Project team will also be responsible for verifying the final deliverables and provide sign-off.

IT requirement will be identified and intimated to the CGDA head quarters for incorporation into IT roadmap. Head quarters IT team shall constitute the Project team (refer Annexure B for sample) to execute the project by following an appropriate methodology which may be strictly followed. All applications must be reviewed at predetermined checkpoints of the software development life cycle (SDLC) by the Application Architect or their designate. Any deviations must be reported and corrective actions determined. Below picture gives a high level view of process that must be followed for each project right from initiation to maintenance/ sustainment.

There shall be a separation between the production, development and test environments. This will ensure that security is rigorously maintained for the production system, while the development and test environments can maximize productivity with fewer security restrictions.



This policy is effective immediately and supersedes all earlier IT policies issued by CGDA. Project teams are required to comply with this policy, policies and guidelines issued by Government of India and

Ministry of Defence. Each project team is expected to follow the checklists mentioned at Annexure A, Project methodology and technologies being used may require additional checklists to be answered.

Requirements Checklist:

- Was thought given to the system administration functionality?
- Was thought given to error handling?
- Does the specification clearly divide the project into phases?
- Do all the phases have verifiable (and preferably undisputable) outcomes?
- Does the document refer to any related documents as specifically as possible? (Document title, revision, page number)?

If there are interfaces:

- Have the necessary data required for interfacing been identified?
- Is the maximum load (data and system usage) estimated?
- Are the security requirements specified?
- Are the operation and maintenance requirements specified (service transition)?
- Has there been a peer-to-peer review (walkthrough)?
- Has the application architect reviewed (walkthrough)?
- Have the requirements/specifications been agreed to and signed off by the user?
- Have reporting requirements been clearly identified?

Design Checklist:

- Is the design as simple as it can be?
- Are all the functions/features that are listed in the requirements covered?
- Are all assumptions, constraints, design decisions and dependencies documented?
- Have all reasonable alternative designs been considered, including not automating some processes in software?
- Does the design have features or functionality which were not specified by the requirements (e.g., are all parts of the design traceable back to requirements)?

- Does the design create reusable components if appropriate?
- Are modules well-defined including their functionality and interfaces to other modules?
- Interface details:
 - Routine name, parameters and their types, return type, pre- and post-condition, usage protocol with respect to other routines
 - File name, format and permissions
- Are all major data structures described and justified?
- Are major data structures hidden with access functions/procedures?
- Is the database organization and content specified?
- Are all key algorithms described and justified?
- Are all major objects described and justified?
- Is the user interface modularized so that changes in it won't affect the rest of the program?
- Is a strategy for handling user input described, i.e., file input, manually entered through filters, etc.
- Are key aspects of the user interface defined?
- Are space use estimates and a strategy for space management described and justified?
- Is a strategy for handling I/O described and justified?
- Is a coherent error-handling strategy included?
- Are error messages managed as a set to present a clean user interface?
- Are necessary buy vs. build decisions included?
- Is this designed to accommodate changes/enhancements in future?
- Is any part over- or under-designed?
 - Are the major system goals clearly stated?
 - Does the complete architecture fit together conceptually?
 - Is the top-level design independent of the machine and language that will be used to implement it?

- Are you, as a programmer who will implement the system, comfortable with the design?
- Design review and Walkthrough completed with architects (data and application)?
- Instructions/documentation for transition to Operations team?

Coding Checklist:

- Does every input that comes from an untrusted source (i.e., typing into fields on a page, external systems) have associated error checking accounted for?
- Are all forms of validation done on the server side? (only allow on the client side on an as needed basis)
- Stored procedures used as the method for data validation/delivery
- Is each coding module of sufficient size? Limit the size for readability and maintainability. Use a 4 page rule of thumb. If the module is larger than 4 pages consider if it can be reduced in size or functionality can be split out; also consider the following:
 - Size and quantity of data that would need to be passed between routines
 - Number of temporary tables that would be needed
 - Any extra database reads/writes that would be required.

Does the code have the following:

- Proper naming convention
- Purpose is documented
- Brief description
- Original author and date are identified
- Change control area showing date of change, reason, person who did it and associated project or ticket number
- Sample execute
- Unit test documented and repeatable
- Sufficient commenting exists throughout the code to make it readable and understandable (i.e., maintainable) and the comments match the actual code.

Testing Checklist:

- Accessibility for disabled requirements have been tested (If applicable)
- Test case(s) have been identified
- Pass/Fail criteria has been identified for each test case
- Page/Report filters
- Data entered in filters is trimmed
- Each valid option is tested for each filter
- Invalid data entered to test error control
- Test security by changing roles

Documents Checklist:

- Document explaining the functional specifications.
- Document on outstanding issues both technical and functional.
- Document on planned technical and functional up-gradations.
- Document explaining the security standards that have been followed and any known issues.
- Document explaining the system sizing related to peak processing power requirement, data storage requirement, data retention requirement and memory requirement.
- Document explaining the system access rights and data confidentiality/ privacy.
- Document explaining logging requirement and how and what logs have been maintained.
- Document explaining performance and environmental conditions under which software has to perform.
- Document explaining data structure, definition, tables and database requirements.
- Document on testing/ accepting quality and performance.
- Document on coding specifications.
- Document explaining directory structure.
- Architectural design document.

- Document explaining detailed design including following.
 - Tasks, reports, classes, functions, procedures, modules.
 - Identify common modules.
 - Define control logic for each task.
 - Identify database and access requirement.
 - Identify internal and external integration points/ interfaces and data exchange.
 - Identify exports and outputs from modules/ system.
 - Logic description.
 - Define exception handling.
- Document explaining code with comments that can be understood by new users.
- Document explaining software dependencies.
- Document explaining development, test, training, production and recovery environment.
- Document on version history.
- Document explaining hard-coding and portability.
- Document explaining Installation procedure (step by step).
- Support manual answering FAQs and errors.
- User manual.
- Document explaining backup and recovery process.
- Document explaining system administration functionality.

Each document must try to convey information in a form other than text and prose use elements like:-

- Structured writing (with heading, small blocks, list, sequences, etc).
- Flowcharts to show graphically the structure and the behavior of the product.
- Tree and graphs to show the structure of the data.
- Lots of Examples.
- Plenty of pictures and screen dumps to aid better understanding.

