OPEN DAGIR FRAMEWORK

CDAO

Guide for Vendors and Practitioners

Vision

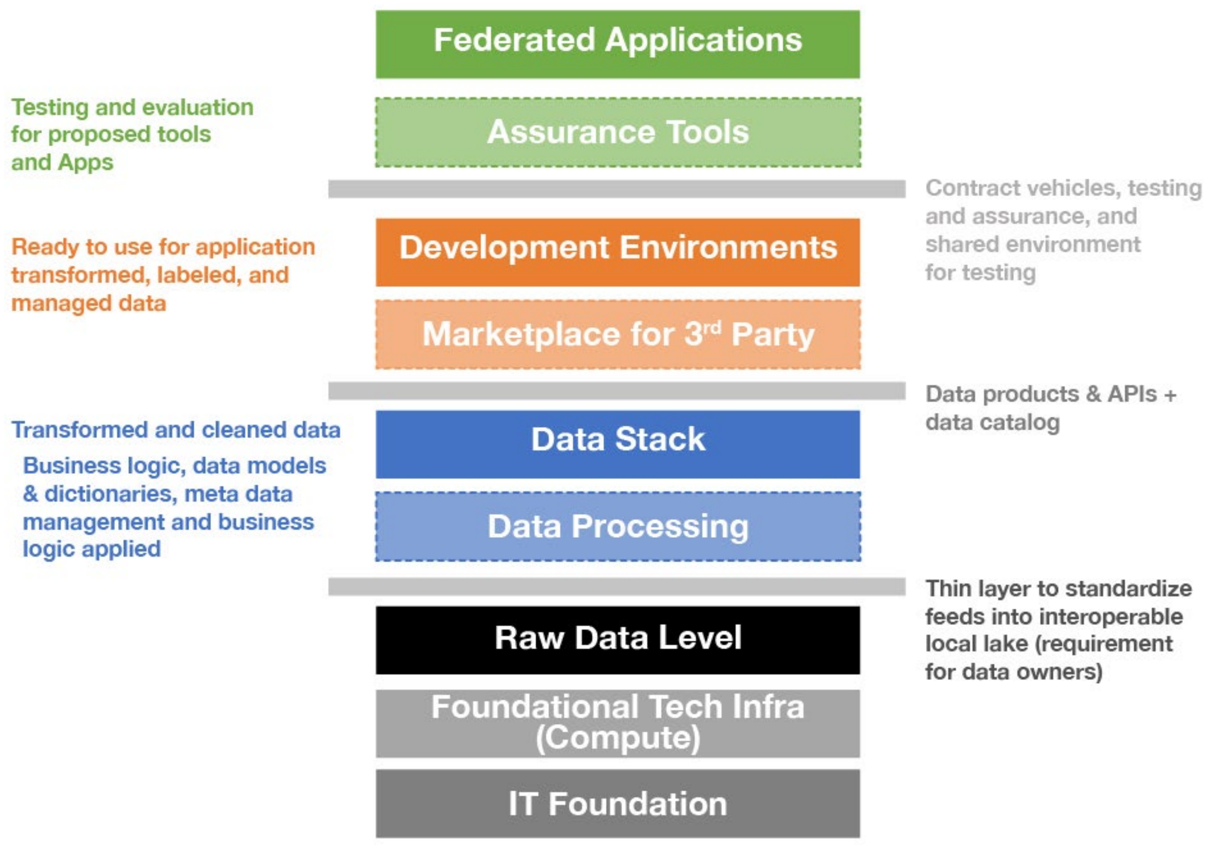
The Government should be able to prototype and adopt new software and data tools quickly, at the speed of mission relevance. That requires being able to prototype and adopt new systems without requiring any modifications of or privileged access to existing systems, and it requires being able to freely share the data the new systems may need. DoD software and data should be readily discoverable, accessible, and interoperable.

In spring 2024, the CDAO rolled out the Open Data and Applications Government-owned Interoperable Repositories (DAGIR) framework to operationalize these principles in scaling data analytics and AI capabilities across the Department of Defense.

What is Open DAGIR?

Open DAGIR is a framework to describe how the Government should acquire software and systems at the speed of relevancy. It has three main components: minimizing the time between identifying a problem and evaluating potential solutions in production with real users and real data; providing clear pathways for proven solutions to scale up to the entire Department; and shaping requirements so that today's acquisitions don't slow us down in the future.

Open DAGIR describes how software systems should be designed and acquired so that functions at one layer of the stack do not require the Government to use the same vendor or system at higher or lower levels.

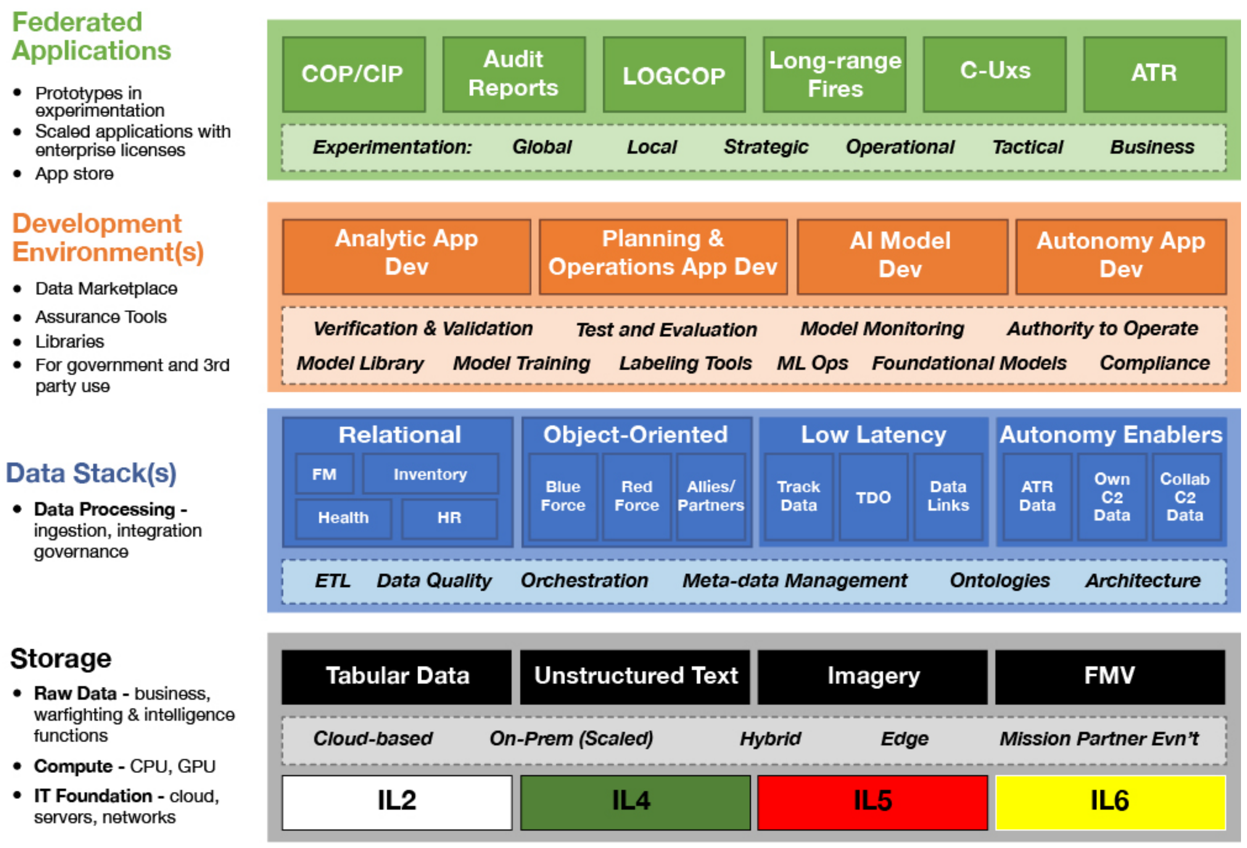

Federated Applications
Testing and evaluation for proposed tools and Apps
Assurance Tools
Contract vehicles, testing and assurance, and shared environment for testing
Ready to use for application transformed, labeled, and managed data
Development Environments
Marketplace for 3rd Party
Data products & APIs + data catalog
Transformed and cleaned data
Data Stack
Business logic, data models & dictionaries, meta data management and business logic applied
Data Processing
Thin layer to standardize feeds into interoperable local lake (requirement for data owners)
Raw Data Level
Foundational Tech Infra (Compute)
IT Foundation

For example, if we contract with vendor to provide a compute platform with developer environments, we do not want to be required to buy any application that runs on that platform from the same vendor. Application development must be open to Government teams working independently and to third party vendors.

If the government contracts with a vendor to retrieve data from a source system and make it sensible to users, we do not want to establish a bottleneck on that data. We do not want to have to pay for it every time we use it, and we do not want to have to recreate the integration for every system that needs access to it. The Government should choose to pay up front for full control of and unlimited access to data created on its behalf, and as much as possible, for any ETL code that produced it. Initiatives to try something new should not be slowed down by trying to find budget or approval for use of existing systems.

Making Open DAGIR Operational

There are a least 4 major levels of the software application Stack where Open DAGIR provides immediate guidance. This list may expand.



- **Federated Applications and Services** – Capabilities in this category are designed to assist mission owners and warfighters in achieving their mission objectives. We want end users to be able to create their own applications, or to buy them from third party vendors, with minimal friction or restrictions from the platforms those applications run on. In particular, developers should be able to use the ordinary general-purpose programming languages of their choice, and they should be able to test and iterate on their applications without coordinating with the platform.

- **Development Environments** – Capabilities in this category are designed for researchers and developers to facilitate their development and maturation process of applications for mission owners and warfighters. Examples of these capabilities are development environments, CI/CD environment, etc. Which platform you use should not dictated by what data the platform owners have access to. Users should be able to access their data equally on any platform.

- **Data and DataOps** – Capabilities in this category are designed to support the productization and discovery of data and data products by providing capabilities such as managed storage solutions, authentication and authorization services, federated data catalog, data mesh services, etc. How data is transformed and made available should not be restricted by the platform those transformations are running on. Users should be able to use general purpose tools and not be limited to the built-in methods integrated into an individual platform. That way, if there is a new capability they require, they can acquire a best of breed solution themselves instead of negotiating with the platform to implement their own version.

- **Computational Infrastructure** – A computation environment that provides application developers computing resources on which to develop and run their capabilities. Developers are able to open standards and resources for development and are not limited to using vendors' proprietary standards, resources, etc.

Within each of the capability categories there are enterprise services, tooling, and contract vehicles to facilitate the development and deployment of capabilities within that category. Enterprise services are also available to provide test and evaluation and/or monitoring of deployed applications to ensure compliance standards are within thresholds. Examples of these services might include: labeling services within the DevOps and MLOps category are available to provide labeled data for AI/ML model researchers and developers; or workflow orchestration tools within the Data and DataOps category that are available to streamline the development and maintenance of complex data pipelines.

Enterprise services, libraries, etc. are also available to facilitate the exchange of relevant information between capability categories. These services, libraries, etc. provide an abstraction layer to the specific capability implementation. For example, application developers will leverage enterprise REST API endpoints to retrieve relevant data for their application. This approach eliminates the need for application developers and providers to know what type of data store is used, how the data is stored, etc. and eliminates the need to write application-specific logic to access data from a specific data store. The abstraction allows the flexibility of changing the underlying implementation without impacting the end-user. Ultimately, there is a bidirectional benefit with this approach: Developers will develop against standard APIs for their dependencies without needing to know the underlying implementation of their dependencies and this will allow capabilities to be interoperable and not be limited to a specific approach or implementation.

Technical Principles

- Create **modularity** by requiring major components to interact via open, defined patterns or protocols. DoD systems that are tightly coupled between the data layer, the application layer, the hardware layer, etc. cannot readily change one component part of the tech stack for another without replacing additional component parts. And they cannot switch vendors or experiment with a novel approach without major unrelated technical work.

- Ensure **interoperability** by requiring APIs and dependencies to be expressed via open standards. For example, applications might be required to encapsulate their dependencies in containers and be managed at runtime by a framework like Kubernetes. APIs might be required to be documented according to OpenAPI or a similar specification. DoD applications that did not conform to open standards would prevent the Department from being able test and evaluate similar capabilities and slow the adoption of best of breed approaches.
- **Single Responsibility** – Each module should only have one responsibility. Modules with two should be split up. Updating modules that multiple responsibilities will have a higher risk of introducing undesired artifacts due to tightly coupled dependencies / logic and lack robust testing.
- **Dependency Inversion Principle** – High-level modules should depend on high level interfaces / abstractions and not specific implementation of a particular dependency. Modules that have dependencies on specific implementation of other modules will suffer from unexpected failures if the dependency is ever replaced. Whereas if a module is dependent on an abstraction or high-level interfaces, this will enable changing / updating dependency modules in a transparent manner.
- **Data is Shared** – Data is discoverable and accessible in a timely manner where ownership, maintenance and governance are managed by the government.

Open DAGIR will consist of a set of **wrappers**, messaging patterns, and enterprise core services to allow applications to run in an interoperable manner while data and data products are readily available via defined access patterns. Open DAGIR platforms will support and/or implement the following but not limited to:

- APIs and interfaces as defined by CDAO
- Messaging patterns and/or protocols as defined by CDAO
- Make use of core enterprise services provided by CDAO

Open DAGIR Implementation

The ability to share and access data across systems, tools, and user groups utilizing a standard interface and structure is critical in supporting rapid development in an open environment. These aims are already being realized through core CDAO efforts like Enterprise APIs, the DoD Data Catalog, Perceptor, and a modern data and model marketplace achieved through integrating these products. While these capabilities are currently deployed on NIPR, CDAO plans to deploy to higher environments in 2025.

The Enterprise API layer has been deployed on NIPR for over two years, making data products easily accessible across DoD. As part of the deployment, a developer portal was also created to enable ease of API discovery. Outside of solely serving data and data products, the Enterprise API also provides a means to interface with deployed applications. Interested parties can utilize metadata output, check status, or query into an application. The Enterprise API publishes a standard set of endpoints that can be used across the ecosystem, providing cohesion across disparate tools and systems as well as a familiar user and developer experience.

The DoD Data Catalog serves as a primary data discovery tool ensuring the Department's data assets are broadly accessible, visible, understandable, and trustworthy. The CDAO established a community of practice with data stewards across the Department, however this scope has expanded beyond the DoD, as the DoD Data Catalog has initiated metadata sharing with external federal agencies such as the Department of Veterans Affairs, Central Intelligence Agency, and the Department of State.

The DoD Data Catalog drives open standards and accessibility for metadata sharing, creating a consistent experience to discover and access DoD data. To enable more scalable self-service capabilities, integration efforts between the DoD Data Catalog and the Enterprise API are underway to provide standard endpoints allowing data stewards and systems to push and pull metadata to and from the DoD Data Catalog. These efforts serve as the foundation for a data marketplace that enables data consumers to seamlessly discover and access data products with high quality and reliability.

Conclusion

CDAO has made important progress since the announcement of the Open DAGIR construct in spring 2024, and will continue to put these principles into action in 2025. Companies with innovative solutions can submit pitch videos on CDAO's Tradewinds Solutions Marketplace. Stay tuned for further details as DoD continues to make progress in scaling data analytics and AI capabilities.