Glencore Development Notes

Jonny Coombes

November 2020

Contents

Chapter 1

General Notes

1.1 Versioning

The general versioning scheme for Argon builds will comprise of three main components:

1. The *major* version number.
2. The *minor* version number.
3. Either:
   (a) A specific build identifier (monotonically increasing).
   (b) A specific patch/correlated fix identifier.

1.2 Source Control

The Git versioning scheme for Argon is straightforward:

1. The main development branch is *mainline*.
2. Each intermediate release will have a specific branch, named in accordance with the release. Currently planned intermediate releases are as follows:
   (a) 0.1.0 - build after the initial development sprint.
   (b) 0.2.0 - build after the second development sprint.
   (c) 0.3.0 - build after the third development sprint.
3. Individual feature implementations will be carried out on a dedicated branch, prefixed with the corresponding **Jira** ticket. For example, ticket number **JA-15** would have a branch named *feature*
   JA-15-Summary, where the summary is automatically generated as part of the development toolchain.
4. Within the local development environment, changes are mastered and then pushed to multiple remotes. (There may be multiple remotes based on the number of environments stood up).

1.3 Build Dependencies

The key libraries used throughout the build of the Argon project as given in table 1.1 below:

Library	Version	Description
.NET Core	5.0	Core .NET platform runtime
ASP.NET	5.0	ASP.NET Core library
EF Core	5.0	EF framework (plus RDBMS specifics)
Serilog	2.10.0	Logging library
Serilog.Sinks.Console	3.1.1	Console sink for Serilog
Polly	X.X	Policy library

Table 1.1: Key Argon Dependencies

1.4 Database Notes

General DB notes:

1. **Development SQL Server Version (Ryleh)**: Microsoft SQL Server 2019 (RTM) - 15.0.2000.5 (X64) (Sep 24 2019 13:48:23)

2. **Development Argon Login Name** : *argon*

3. **Development Argon Db Name**: *argon*

4. **Core Schema Name**: *core*

The nominated user used to connect through via the DB context, must be a member of the following SQL Server roles:

1. dbcreator

This allows for just a user login to be allocated on the target SQL instance, and then Argon can take care of creating the necessary database/schema objects. (*This adheres to the principal that configuration should be a close to zero as possible for new deployments*).

1.4.1 Core Tables

Schema	Table	Description
core	collection	The main collections table, one row per collection. A *collection* may have an associated *constraintGroup*, however there is no stipulation that a given constraint group can't be shared across multiple collections.
core	constraintGroup	The main aggregate table for constraints
core	constraint	The collection constraints table. $1 \rightarrow *$ relationship between entries in the *constraintGroup* table and entries within this table.
core	item	The collection items table. $1 \rightarrow +$ relationship between the *collection* table and entries within this table.
core	version	The collection item versions table. $1 \leftarrow +$ relationship between entries in this table and the items table. (Each item will have a minimum of one version).
core	propertyGroup	The main property group table there is a $1 \rightarrow *$ relationship between this table and the property table. A given item or collection may have a property group associated with it, however the underlying schema also allows for property groups to be *shared* between multiple items and even collections.
core	property	The property table. Each item (and potentially) collection will have multiple associated rows within this table, through a given property group link.

Table 1.2: Core Tables

1.4.2 EF Core Specifics

Some general notes on the EF Core implementation within Argon:

1. All useful model entities will utilise a *Timestamp* as a concurrency token.
2. All internal **Guid** entries that aren't relevant to the responses have been squelched through configuration of the Json serialisation layer.
3. Null values (for nullable types that is) have also been stripped out of response generation through configuration of the Json serialisation layer.

1.4.3 OTCS Specifics

General notes on the dev/test versions of OTCS (Opentext Content Server):

1. Currently baselined version is 16.0.18 (no patches).
2. Need to check that latest patch set doesn't raise any regression issues with the opentextRestful virtual storage provider.

1.5 AWS Deployments

Current AWS deployment details in table 1.3 below: Route53 FQDN Entries:

1. `argon-dev-1.jcs-software.co.uk`

Component	Type	Name	Details
VPC	Virtual Private Cloud	Argon-Dev-VPC	Partitioned VPC
Subnet	VPC Subnet	Argon-Dev-SN-1	10.0.0.0/24 CIDR
Sec. Group	Security Group	Argon-Dev-SG-1	SSH, HTTPS, SQL (Restricted)
Internet Gateway	(E/I) Gateway	Argon-Dev-IG	Ingress and outgress
EIP	Elastic IP	No public DNS yet	34.249.105.124
Host	t3a.small	argon-dev-1	Ubuntu 18.04 LTS
Host	t3a.small	argon-dev-2	Ubuntu 18.04 LTS
Host	t3a.small	argon-dev-3	ubuntu 18.04.LTS

Table 1.3: AWS Deployment Artifacts

1.6 Code Notes

1.6.1 Namespaces

The general layout of the Argon core code adheres to the following conventions:

1. The top level namespace for the core is **JCS.Argon**
2. Key code artifacts are organised so that:
   (a) Controllers are placed in the **JCS.Argon.Controllers** namespace.
   (b) Services are placed in the **JCS.Argon.Services** namespace.
   (c) Model elements have a top-level namespace of **JCS.Argon.Model**.

1.6.2 OpenAPI Endpoints

The OpenAPI specification that the API conforms to is published at the following location (relative to the deployment root):

```
/swagger/v1/swagger.json
```

The version path component is expected to remain constant during the initial release, and will only change with *significant* breaking change releases in the future. All initial API endpoints will be prefixed as follows:

```
http[s]://[host]:[port]/api/v1
```

§

Cloud Hosted Endpoints

During development instances of the current build/deployed artifacts are available at the following locations:

```
http[s]://argon-dev-1.jcs.software.co.uk/api/v1
http[s]://argon-dev-1.jcs-software.co.uk/swagger/v1/swagger.json
```

Chapter 2

Release Notes

2.1 Build 0.1.15

Deploy Date 23rd November, 2020

This release represents the first stable build of the Argon core with a limited feature set. Within this release:

1. The core database schema is largely complete, however still in a state of flux, given that complete implementation of the constraint and property group abstractions is not yet complete.
2. The core interfaces around the *VSP* subsystem are defined and 95% implemented for the *Native File* VSP provider. (Still subject to change throughout sprints 2 and 3).
3. The general API surface area and associated Swagger definitions is approximately 95% complete.
4. Asynchronous background processing for item checksums is not yet implemented.
5. Not authentication is present within this build.

2.2 Build 0.2.17

Deploy Date 25th November, 2020

This release is the build at the end of Argon Sprint 2. A number of issues to mention:

1. The currently baseline version of OTCS needs upgrading to 16.0.18 and a patch list should be issued for CS16.
2. Attribute and category ID caching still needs to be implemented (huge optimisation).
3. A cache of cats/atts can be built up through a *query?* across subtype 130 nodes.
4. Need to validate the build against Windows and Ubuntu variants of OTCS.
5. Virtual storage provider implementations are now dynamically loaded and scanned during startup/first request.

2.3 Build 0.3.X