Implementing a build manager in Ada

Stéphane Carrez

FOSDEM 2022
Ada DevRoom
Introducing Porion

• Why a new build manager?
  – Jenkins is slow and uses 1.3Gb memory RSS
  – Requires Java on build nodes
  – Regular security vulnerabilities

• Some requirements:
  – Security & perf: safe design in Ada
  – CLI and web interface
  – Flexible build nodes (ssh, docker, virsh,...)

https://gitlab.com/stcarrez/porion
What a build manager must know

- Define projects with source control method
- Define recipes to build the project
- Define project dependencies
- Define build nodes (different systems, CPUs,...)
- Define build information to track build results
- Define build and project metrics
- Store credentials to connect to build nodes
- Store API secret keys to publish
- More secret keys to sign builds…
What a build manager must do

- Probe source changes in projects
- Schedule builds according to changes (management of a build queue)
- Launch builds (locally or remotely)
- Control and track build execution
- Collect build results (coverage, tests, logs)
- Publish build results
- Send build notifications
- Keep managers happy by providing reports
What a build manager must protect

• A build manager has access to sensitive data
• It must protect sources files (proprietary projects)
• It must protect API secret keys
• It must protect credentials (checkout sources, connect to build nodes)
• It must protect the secret keys to sign or to publish
• It must protect build results and build logs
• It must not leak API secret keys through logs
Porion numbers

- Cost: 1.5 engineer month so far (260 hours on my free time)
- 95% Ada, 2% HTML, 0.4% Typescript
- 32K CLOC Ada (16K generated)
- 43 Ada packages, 30 private Ada packages
- 19 database tables
**Architecture**

**Web Server**
- porion-server
  - AWA
  - AWS

**Command Line**
- porion
  - Ada Database Objects

**Filesystem**
- config
- tmp
- logs
- projects
  - Ada Keystore
  - SQLite Database

**Build Nodes**

https://gitlab.com/stcarrez/porion
Porion Agent Architecture

Advanced Resource Embedder

Porion Agent (commands)

Porion Lib

Printer Toolkit

Ada Keystore

Ada Database Objects

XML/Ada

Ada EL

Ada Util

SQLite

NetBSD

FreeBSD

GNU/Linux

Dynamo

https://gitlab.com/stcarrez/porion
Porion Server Architecture

- Porion Web Server
- Porion Lib
- Ada Web Application
  - Ada Servlet
  - Ada Server Faces
  - Ada Security
  - OpenAPI Ada
  - Ada Database Objects
  - Ada Web Server
  - XML/Ada
  - Ada EL
  - Ada Util
  - Ada Keystore

- SQLite
- NetBSD
- FreeBSD
- GNU/Linux

https://gitlab.com/stcarrez/porion
UML to Ada generation

- Described the database model in UML:
  - 19 tables organized in 5 packages
- Used ArgoUML (Java tool):
  - Tool migrated from tigris.org to GitHub
  - Works very well to define the UML class model
- Used Dynamo for the code generation:
  - It reads ArgoUML file
  - It generates Ada model for the UML classes
  - It generates SQL table creation schema
Database Modeling

Design
- UML Model
- YAML Model
- XML Model

Generate
- Dynamo
- Generator
- Model Doc (HTML)
- Ada Model Packages
- SQL Tables

Develop
- Your Application Code
- Generated Application Model Packages
- Ada Database Objects Library
- Ada Utility Library
- Postgresql, MySQL or SQLite

https://gitlab.com/stcarrez/dynamo
UML generated Ada code

- 14K CLOC generated in 6 Ada packages
- Handles SQL insert, update, delete, queries
- Uses Ada.Containers.Vectors for lists
- Reference counting for objects

https://gitlab.com/stcarrez/dynamo
A tour to Porion UML model
Benefit of UML and Ada

- UML database model is not right at the first time
- Several iterations to add new tables, new relations or new attributes in UML model
- Easy and fast generation of Ada from UML
- Changes in UML model breaks the compilation and can be identified and fixed
- Consistency between Ada and SQL
- Refactoring is safe due to Ada strong typing!
Focus: build queue scheduler 1/3

- Role of the build queue and its scheduler:
  - Keep an ordered list of recipes that must be built
  - Minimize the number of builds
  - Take into account project dependencies

Dependencies

Add B in queue

https://gitlab.com/stcarrez/porion
Focus : build queue scheduler 2/3

- **Load the build queue in an Ada vector**

  ```ada
  with Porion.Builds.Models;
  Query : ADO.Queries.Context;
  DB : ADO.Sessions.Session;
  
  Query.Set_Filter ("o.node_id = :node_id");
  Query.Bind_Param ("node_id", Node_Id);
  Porion.Builds.Models.List (Queues, DB, Query);
  ```

- **Compare two build queue entries**

  ```ada
  function "<" (Left, Right : in Build_Queue_Ref) return Boolean;
  ```
Focus : build queue scheduler 3/3

- Instantiate the sort package

```plaintext
package Sort_Queue is
    new Build_Queue_Vectors.Generic_Sorting ("<" => "<");
end;
```

- Sort the build queue vector

```plaintext
Queues.Append (New_Item);
Sort_Queue.Sort (Queues);
```

- Update the queue order and save in the database

```plaintext
declare
    Order : Natural := 0;
begin
    for Queue of Queues loop
        Queue.Set_Order (Order);
        Queue.Save (Service.DB);
        Order := Order + 1;
    end loop;
end;
```
Embedding resource

• Problem:
  - How to configure the database?

• Solution:
  - Embed the SQL schema definition in the binary
  - Have an array of String with each String being an SQL create table statement
  - Use ARE to embed the SQL schema
  - Generates 2K CLOC in 3 Ada packages
Advanced Resource Embedder

Resources
- Config
- Help
- Web

Generate
- Advanced Resource Embedder
  - Rules
    - Copy
    - Minify
    - Compress

Build
- Compiler (C, Ada, Go)

Run
- Binary Executable
  - Configuration Files
  - Help Files
  - Web Files (HTML, CSS, JS...)

https://gitlab.com/stcarrez/resource-embedder
ARE generated code

- Types are declared in a parent package

```plaintext
package Porion.Resources is

    type Content_Array is array (Natural range <>)
    of access constant String;
    type Content_Access is access constant Content_Array;

end Porion.Resources;
```

- ARE generates a child package with function declaration and static constant array of strings

```plaintext
package Porion.Resources.Schema is

    function Get_Content (Name : String) return Content_Access;

end Porion.Resources.Schema;
```
Conclusion

- Lessons learned:
  - Writing a build manager is hard (secure is harder!)
  - Ada helps by forcing to think more about your design
- Code generation can speed up development:
  - Dynamo: UML => Ada mapping & SQL schema
  - ARE: SQL files => Ada package with static content
- High level database representation is key:
  - Load, insert, update database objects easily
  - Implement complex algorithm easily

https://gitlab.com/stcarrez/porion
Questions