Operating System Architecture Language WG/RG/CG 1st Meeting

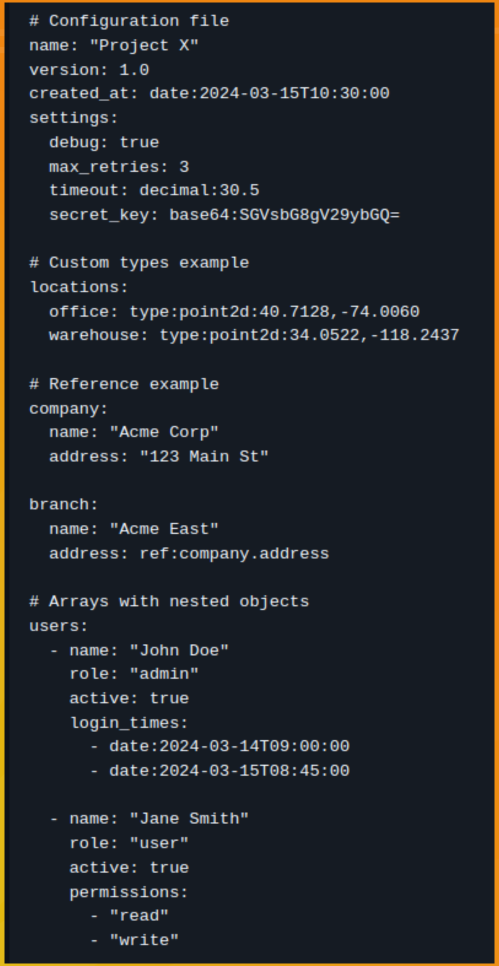
By: Ronaldson Bellande
PhD Student
Founder/CEO/CTO/COO Bellande Technologies Corporation Inc
Founder of Bellande Research Organizations

Meeting Agenda

- Data/File Format
- Operating System Security
- Operating System Permissions
- Scripting Language

Data/File Format

- Bellande File Format has many benefits including; Data Types Support, Structure Features, Data Integrity, Data Processing, Advanced Features, Format Characteristics, Development Features, Export/Import



```
# Configuration file
name: "Project X"
version: 1.0
created_at: date:2024-03-15T10:30:00
settings:
  debug: true
  max_retries: 3
  timeout: decimal:30.5
  secret_key: base64:SGVsbG8gV29ybGQ=

# Custom types example
locations:
  office: type:point2d:40.7128,-74.0060
  warehouse: type:point2d:34.0522,-118.2437

# Reference example
company:
  name: "Acme Corp"
  address: "123 Main St"

branch:
  name: "Acme East"
  address: ref:company.address

# Arrays with nested objects
users:
  - name: "John Doe"
    role: "admin"
    active: true
    login_times:
      - date:2024-03-14T09:00:00
      - date:2024-03-15T08:45:00

  - name: "Jane Smith"
    role: "user"
    active: true
    permissions:
      - "read"
      - "write"
```

Data/File Format

1. Basic Types
    a. Strings (with intelligent quoting)
    b. Integers
    c. Floating point numbers
    d. Booleans
    e. Null
2. Advanced Types
    a. Decimal (high-precision numbers)
    b. Dates and Times
    c. Binary Data (base64 encoded)
    d. File Paths
    e. Regular Expressions
    f. Complex Numbers
    g. Sets
    h. URLs
    i. Timedeltas
    j. Version Numbers
    k. Custom Types (user-definable)
3. Hierarchical Data
    a. Nested Objects
    b. Arrays/Lists
    c. Mixed Nesting
    d. Unlimited Depth
4. References
    a. Internal References
    b. Cross-file References
    c. Circular Reference Detection
    d. Reference Validation
5. Validation
    a. Schema Validation
    b. Type Checking
    c. Pattern Matching
    d. Required Fields
    e. Value Ranges
    f. Custom Validators
6. Security
    a. Built-in Encryption (AES)
    b. Custom Encryption Support
    c. Checksum Verification
    d. Data Integrity Checks
7. Version Control
    a. Change Tracking
    b. Version History
    c. Author Attribution
    d. Modification Timestamps
8. Compression
    a. Built-in Huffman
    b. Compression
    c. Multiple Compression Algorithms
    d. Streaming Support
    e. Chunk Processing
9. Transformation
    a. Custom Type Transformers
    b. Data Filters
    c. Value Processors
    d. Format Converters

Data/File Format

10. Search and Query
    a. Path-based Queries
    b. Pattern Matching
    c. Index Creation
    d. Search Optimization
11. Document Operations
    a. Merging
    b. Diffing
    c. Conflict Resolution
    d. Patch Generation
12. Metadata Support
    a. Document Properties
    b. Field Annotations
    c. Custom Metadata
    d. Tracking Information
13. Syntax
    a. Human-readable
    b. Clean Indentation
    c. Comment Support
    d. Clear Structure
    e. Compatibility
14. UTF-8 Support
    a. Platform Independent
    b. Language Agnostic
    c. Extensible Format
15. Performance
    a. Streaming Parser
    b. Efficient Memory Usage
    c. Optimized Processing
    d. Large File Support
16. Error Handling
    a. Detailed Error Messages
    b. Line Number References
    c. Error Recovery
    d. Validation Reports
17. Debugging
    a. Debug Mode
    b. Verbose Logging
    c. Trace Information
    d. Performance Metrics
18. Format Conversion
    a. JSON Export/Import
    b. YAML Export/Import
    c. XML Export/Import
    d. CSV Export/Import
    e. INI Export/Import
19. Integration
    a. Command Line Interface
    b. API Support
    c. Library Integration
    d. Tool Ecosystem

Operating System Security

Bellande Operating System has different access levels. Each has a different purpose and different levels of access. OWNER/BELL, ROOT, ADMINISTRATION, GROUP, USER.

Operating System Security

OWNER/BELL (Position 1)

- Value: 7 (rwx)
- Calculation: 4(read) + 2(write) + 1(execute) = 7
- Access:
  * All system files and directories
  * Core components
  * Kernel level access
  * Hardware level access
  * Can override all permissions
  * Complete system control

ROOT (Position 2)

- Value: 7 (rwx)
- Calculation: 4(read) + 2(write) + 1(execute) = 7
- Access:
  * System files
  * Configuration files
  * Installation files
  * Startup sequences
  * Cannot access core components
  * Cannot modify kernel

Operating System Security

ADMINISTRATION (Position 3)

- Value: 5 (r-x)
- Calculation: 4(read) + 0(write) + 1(execute) = 5
- Access:
  * Read system configurations
  * Execute administrative tasks
  * Manage users
  * Cannot modify system files
  * No core component access
  * No kernel modifications

GROUP (Position 4)

- Value: 3 (-wx)
- Calculation: 0(read) + 2(write) + 1(execute) = 3
- Access:
  * Modify group files
  * Execute group programs
  * Share within group
  * No read outside group
  * No system modifications
  * Limited to group scope

Operating System Security

USER (Position 5)

- Value: 1 (--x)
- Calculation: 0(read) + 0(write) + 1(execute) = 1
- Access:
  * Execute allowed programs
  * Access own directory
  * Use basic utilities
  * No system modifications
  * No file modifications
  * No read access outside home

Operating System Permissions

77000 - System Critical Files

Owner: 7 (rwx) = 4+2+1 : Full control

Root: 7 (rwx) = 4+2+1 : Full control

Admin: 0 (---) = 0+0+0 : No access

Group: 0 (---) = 0+0+0 : No access

User: 0 (---) = 0+0+0 : No access

Use: Core system files, kernel components

77530 - Administrative Tools

Owner: 7 (rwx) = 4+2+1 : Full control

Root: 7 (rwx) = 4+2+1 : Full control

Admin: 5 (r-x) = 4+0+1 : Read + Execute

Group: 3 (-wx) = 0+2+1 : Write + Execute

User: 0 (---) = 0+0+0 : No access

Use: System management tools, configuration files

Operating System Permissions

75531 - Standard Applications

Owner: 7 (rwx) = 4+2+1 : Full control

Root: 5 (r-x) = 4+0+1 : Read + Execute

Admin: 5 (r-x) = 4+0+1 : Read + Execute

Group: 3 (-wx) = 0+2+1 : Write + Execute

User: 1 (--x) = 0+0+1 : Execute only

Use: Standard applications, user programs

Scripting Language

Command Execution: Run both built-in and external commands.

Variable Assignment and Expansion: Assign and use variables within scripts or interactive mode.

Control Structures: Implement logic flow using if-else statements, while loops, and for loops.

Functions: Define and call custom functions.

File Operations: Perform basic file I/O operations.

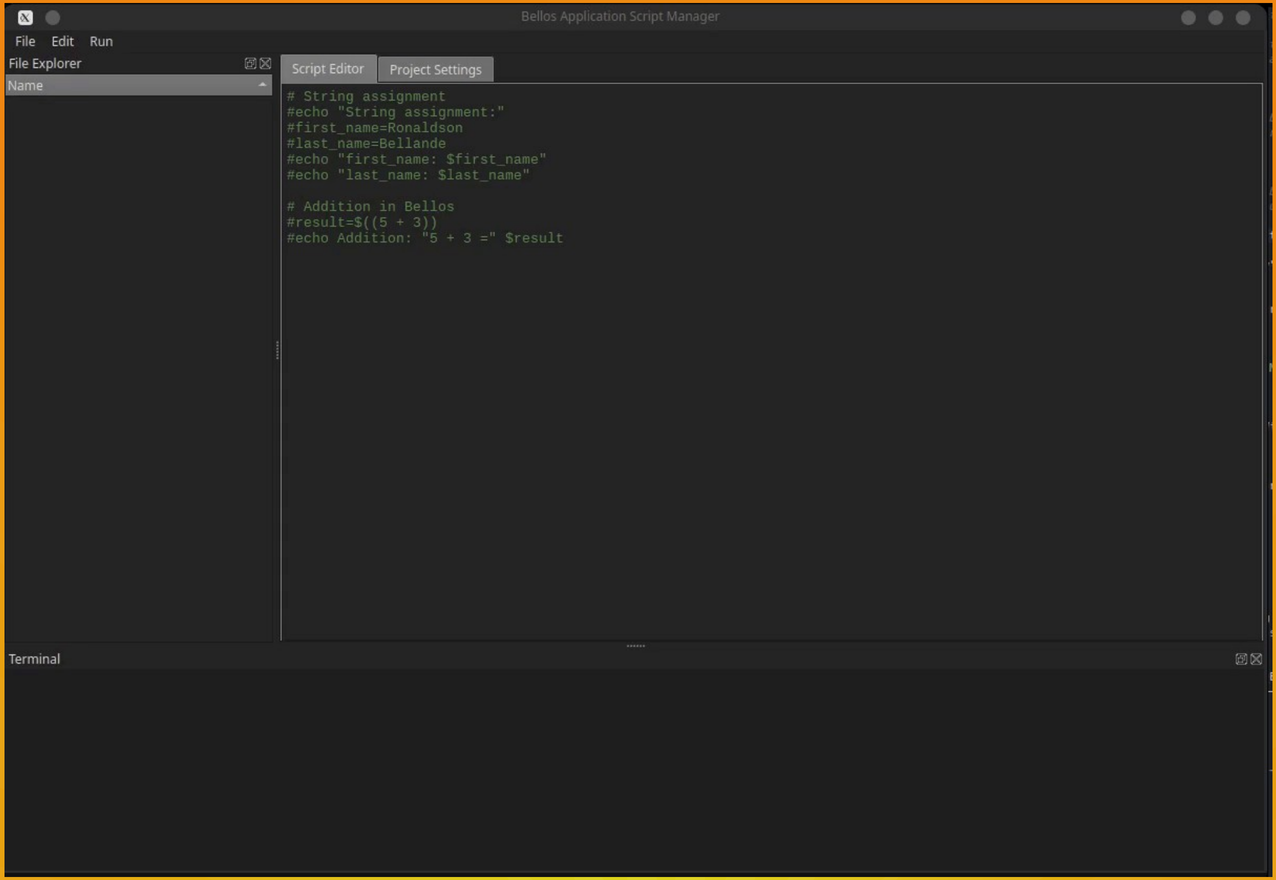
Pipelines: Chain commands together using pipes.

Input/Output Redirection: Redirect command input and output to and from files.

Background Jobs: Run commands in the background.

Environment Variable Handling: Access and modify environment variables.

Scripting Language



```
# String assignment
#echo "String assignment:"
#first_name=Ronaldson
#last_name=Bellande
#echo "first_name: $first_name"
#echo "last_name: $last_name"

# Addition in Bellos
#result=$((5 + 3))
#echo Addition: "5 + 3 =" $result
```

Collaboration Opportunities & Next Steps & Networking & Resources

- Presentation-Notes:https://github.com/Architecture-Mechanism/BAMRI-Operating-System-Architecture-Language-Powerpoint-Notes
- GitHub Organization: https://github.com/Architecture-Mechanism
- Website: https://bellande-architecture-mechanism-research-innovation-center.org
- Discord Group: https://discord.gg/fdkGVKp7wx
- Github Profile: https://github.com/RonaldsonBellande