

ERTS 2020

Practical Application of SPARK to OpenUxAS

M. Anthony Aiello
Claire Dross
Patrick Rogers

Laura Humphrey
James Hamil



ERTS 2020

SPARK & Ada Do Autonomy!

M. Anthony Aiello
Claire Dross
Patrick Rogers

Laura Humphrey
James Hamil



AdaCore R&D?

**Accelerate the development
of new tools & technologies
(e.g., SSI!)**

**Develop use cases and
examples that we can share
with the community**

AdaCore R&D?

**Accelerate the development
of new tools & technologies
(e.g., SSI!)**

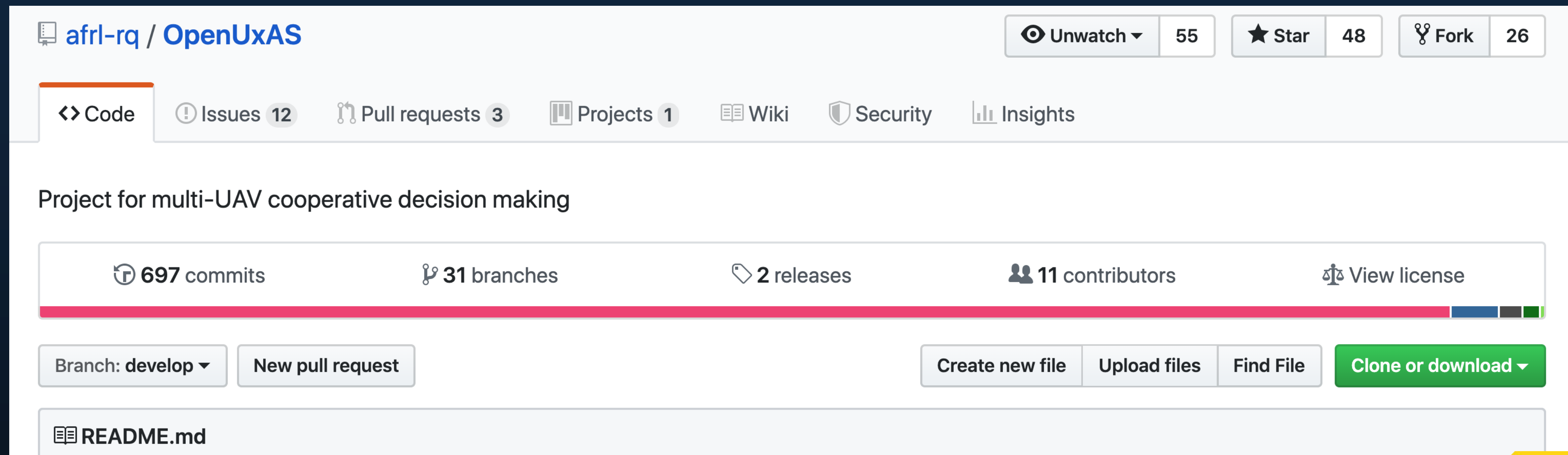
**Develop use cases and
examples that we can share
with the community**



**Our work on
OpenUxAS**

What is OpenUxAS?

An Open Research Platform from AFRL: the US Air Force Research Laboratory



The screenshot shows the GitHub repository page for `aftrl-rq / OpenUxAS`. At the top, there are buttons for `Unwatch` (55), `Star` (48), and `Fork` (26). Below these are tabs for `Code`, `Issues` (12), `Pull requests` (3), `Projects` (1), `Wiki`, `Security`, and `Insights`. The repository description is "Project for multi-UAV cooperative decision making". Below this, a bar shows statistics: 697 commits, 31 branches, 2 releases, 11 contributors, and a `View license` link. At the bottom of the repository view, there are buttons for `Branch: develop`, `New pull request`, `Create new file`, `Upload files`, `Find File`, and `Clone or download`. A `README.md` file is listed below the buttons.



License



Current UAV Operations

Currently, multiple people operate a single UAV

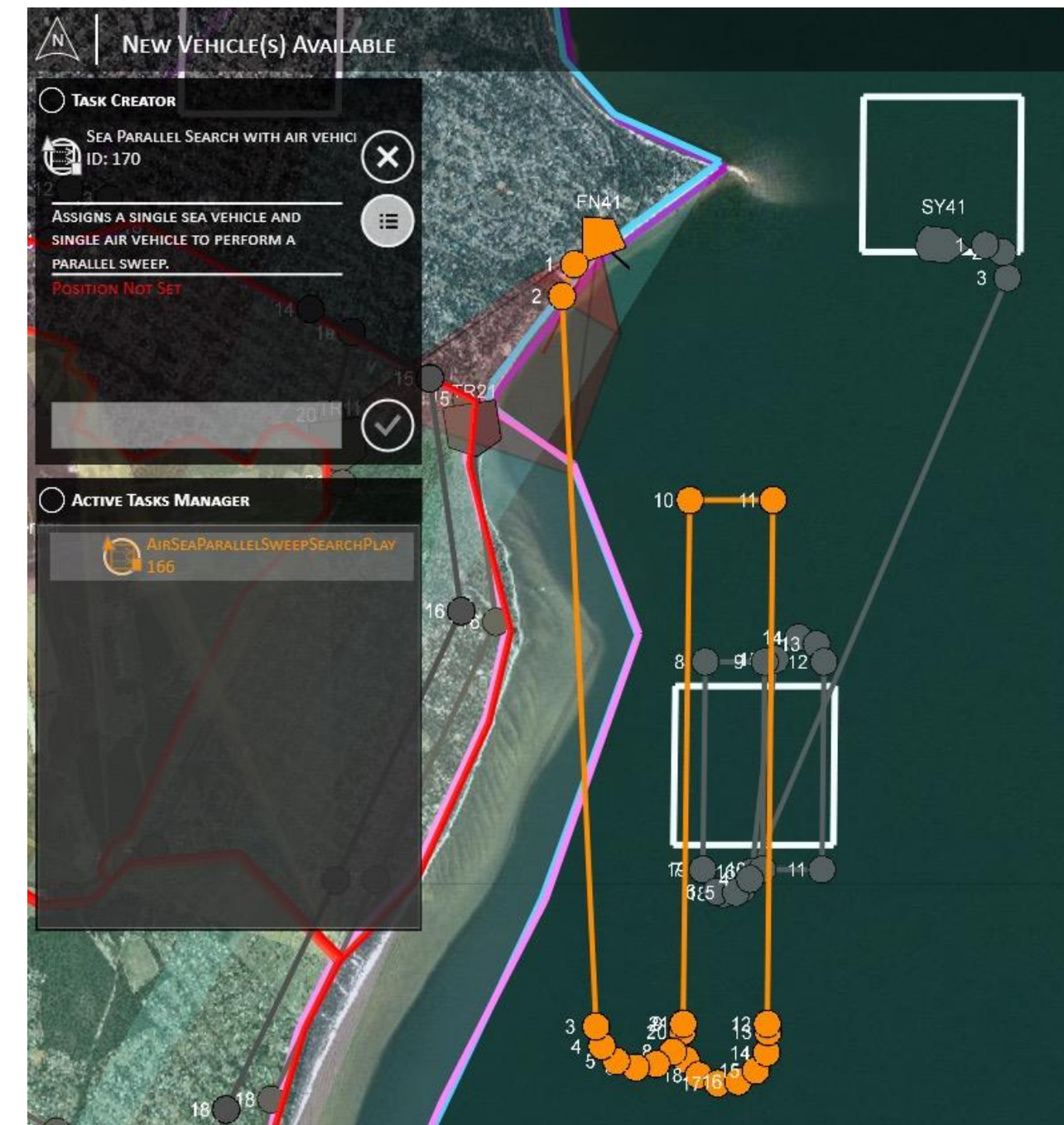
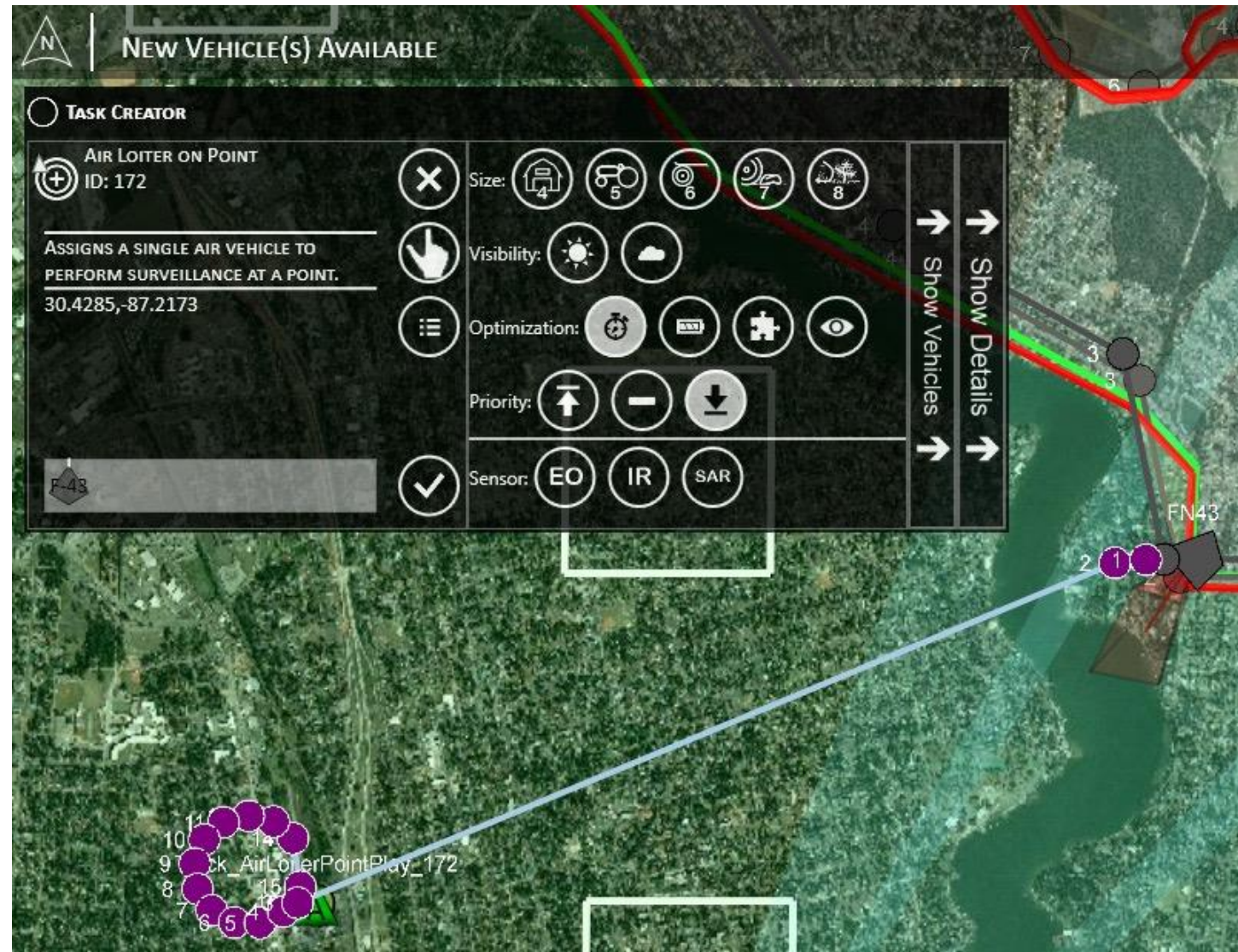
- **pilot**
- **sensor operator(s)**
- **supervisors to oversee & coordinate**



source: nasa.gov

Future UAV Operations

One operator + autonomy software control multiple UAVs



Future UAV Operations

Operator: marks mission objectives & keep-out zones

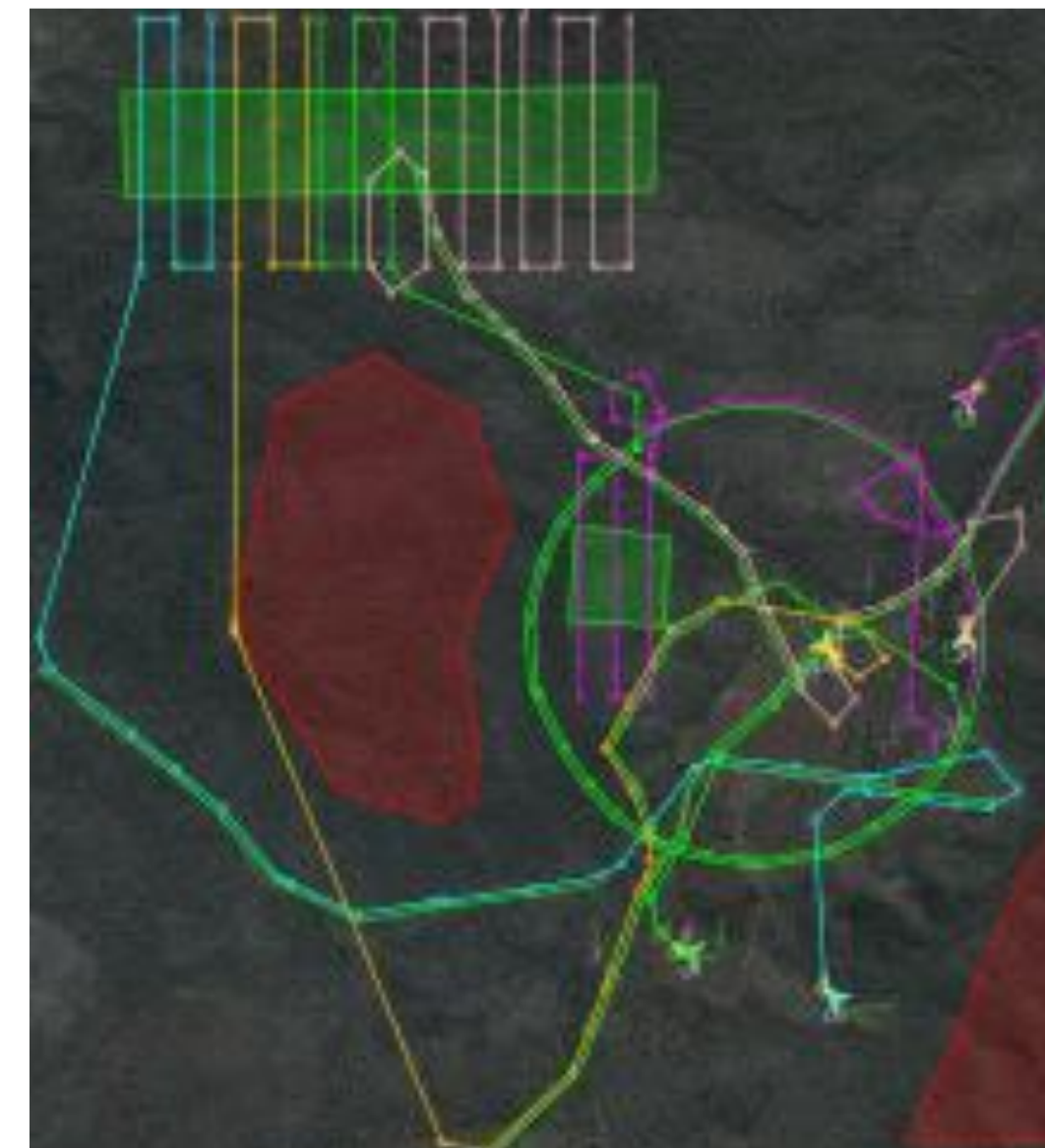


Future UAV Operations

Operator: marks mission objectives & keep-out zones

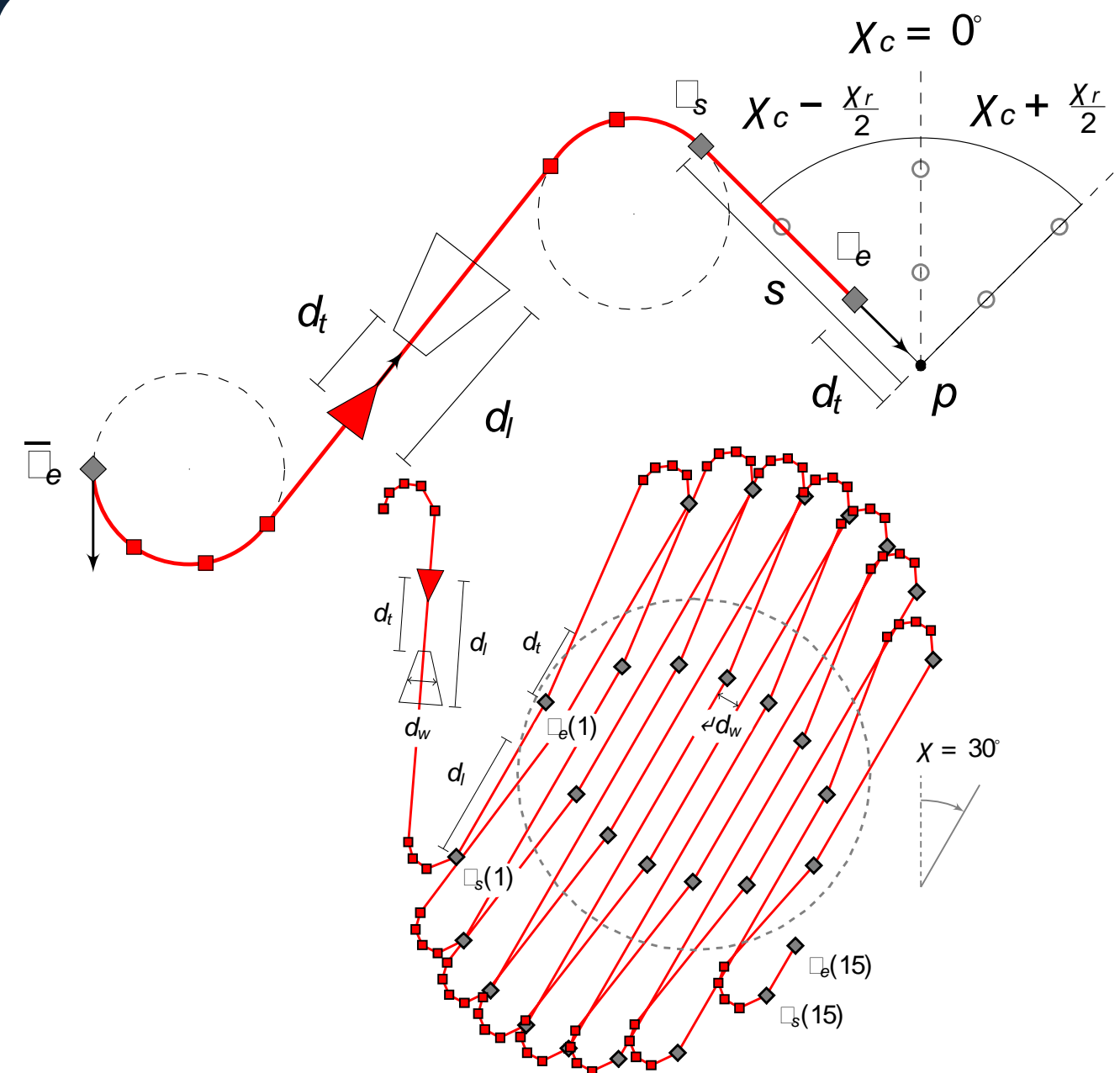


Autonomy: plans UAV flight paths & sensor steering



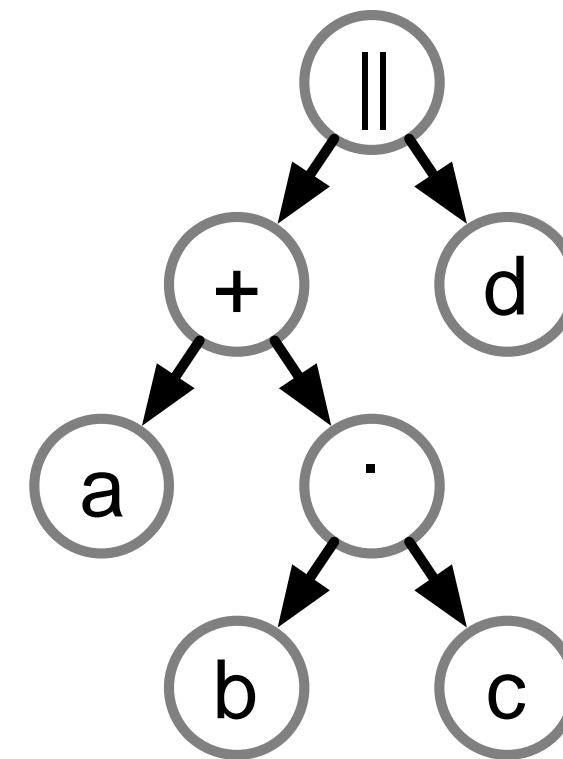
What is OpenUxAS? — User View

Autonomous, Cooperative Control of Multiple Assets



Tasks

$$(a+b \cdot c) \parallel d$$



Process Algebra



Task Assignment

What is OpenUxAS? – Developer View

A Service Oriented Architecture



AFRL's Question

**Can We Show Functional Correctness
of OpenUxAS?**

Why Functional Correctness?

Keep-out zones may represent

- **Terrain**
- **Obstacles**
- **Mission no-go areas**

**Keep-Out
Zone**



Why Functional Correctness?

Keep-out zones may represent

- **Terrain**
- **Obstacles**
- **Mission no-go areas**

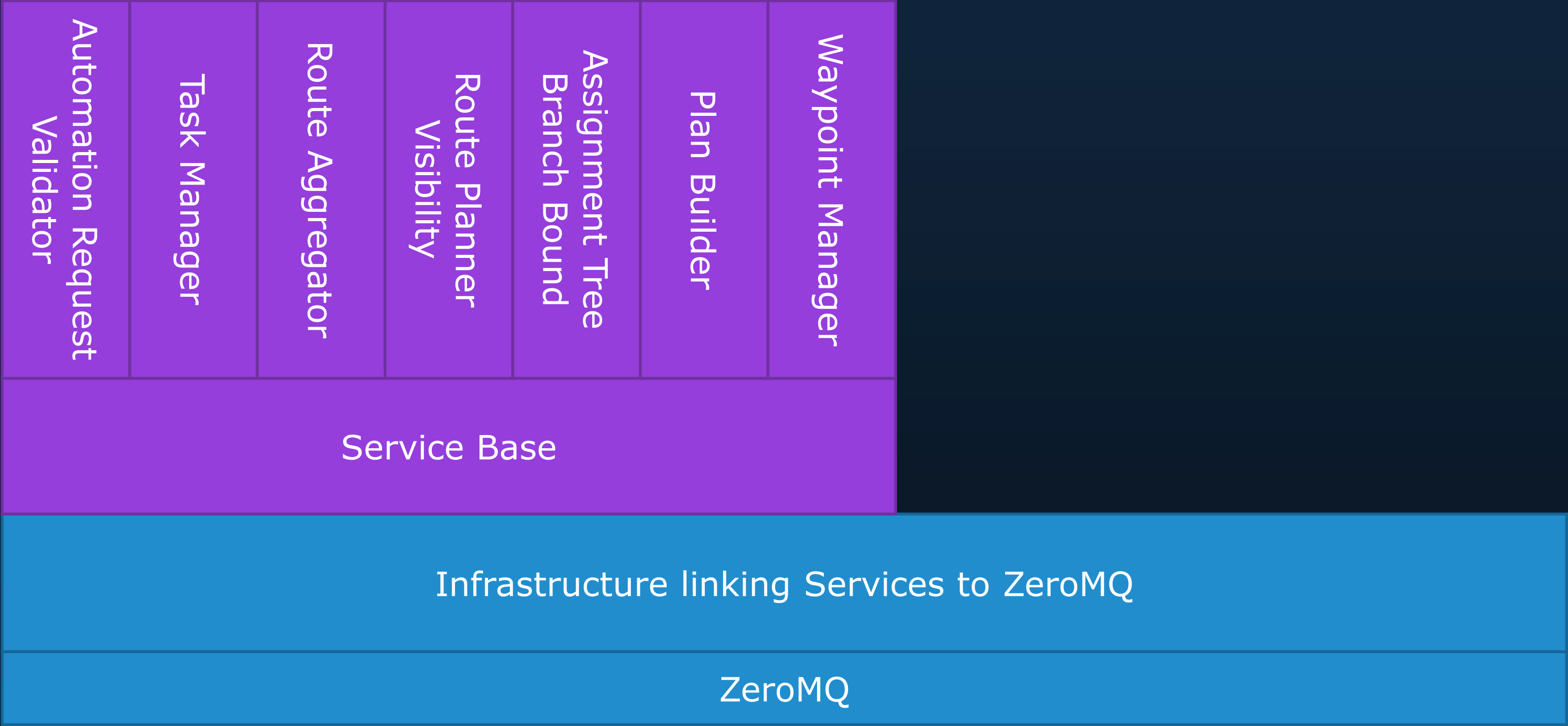
**For mission, privacy, or safety reasons,
avoid keep-out zones.**

**Keep-Out
Zone**

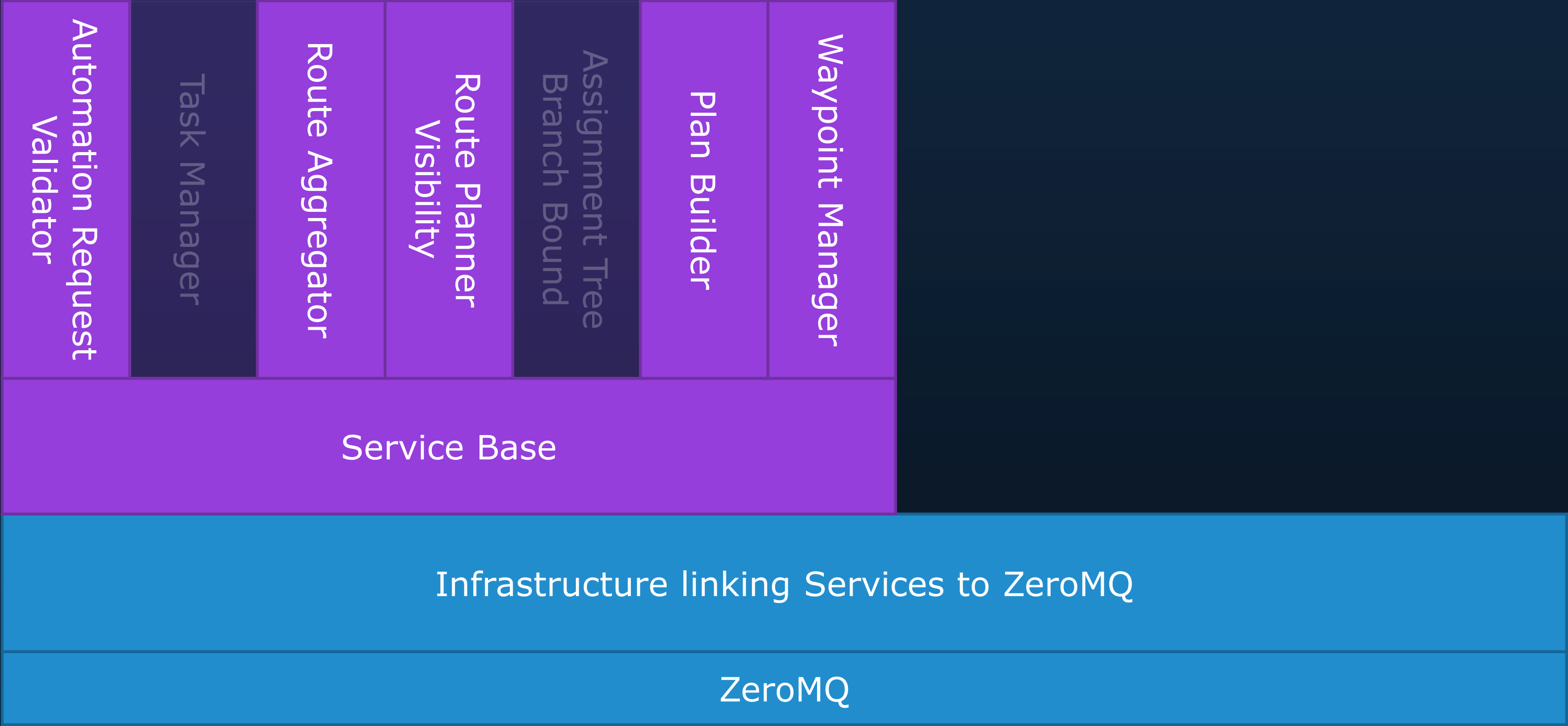
**Planned
Flight
Path**



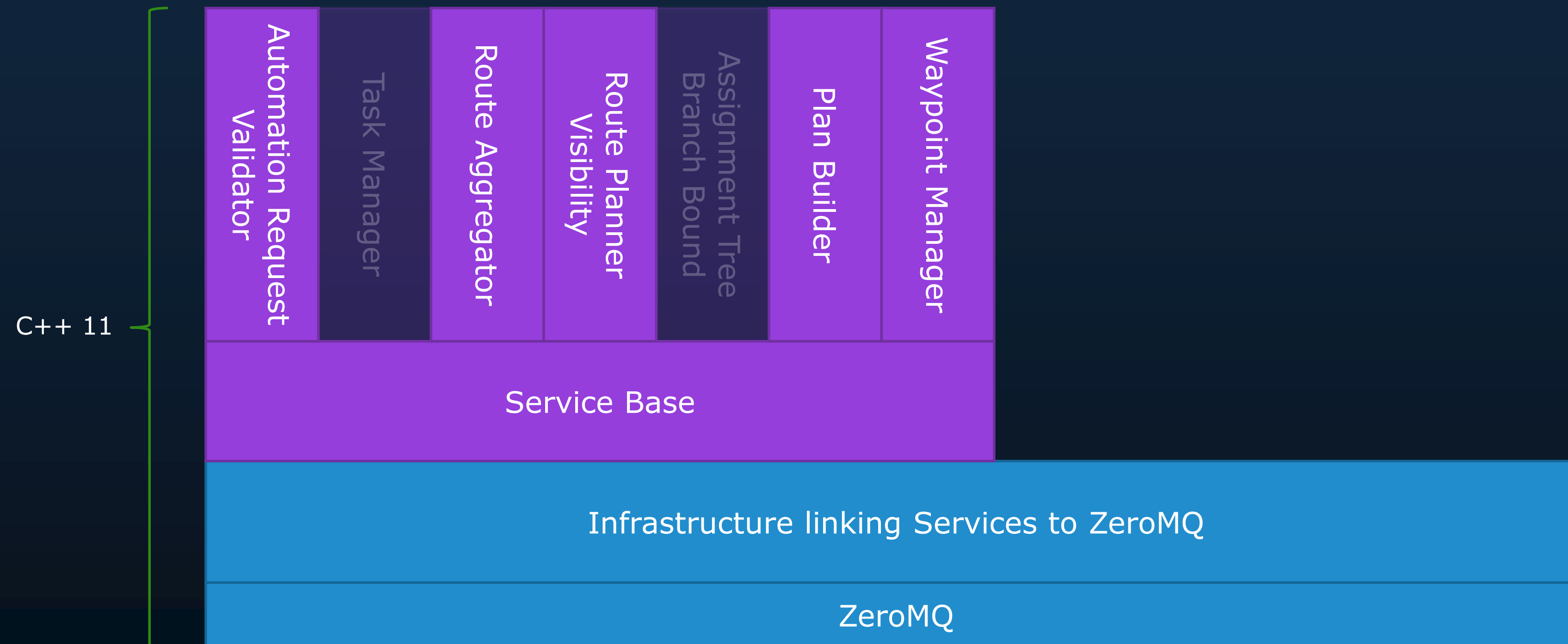
OpenUxAS Components



Critical for Functional Correctness



How Was OpenUxAS Originally Built?



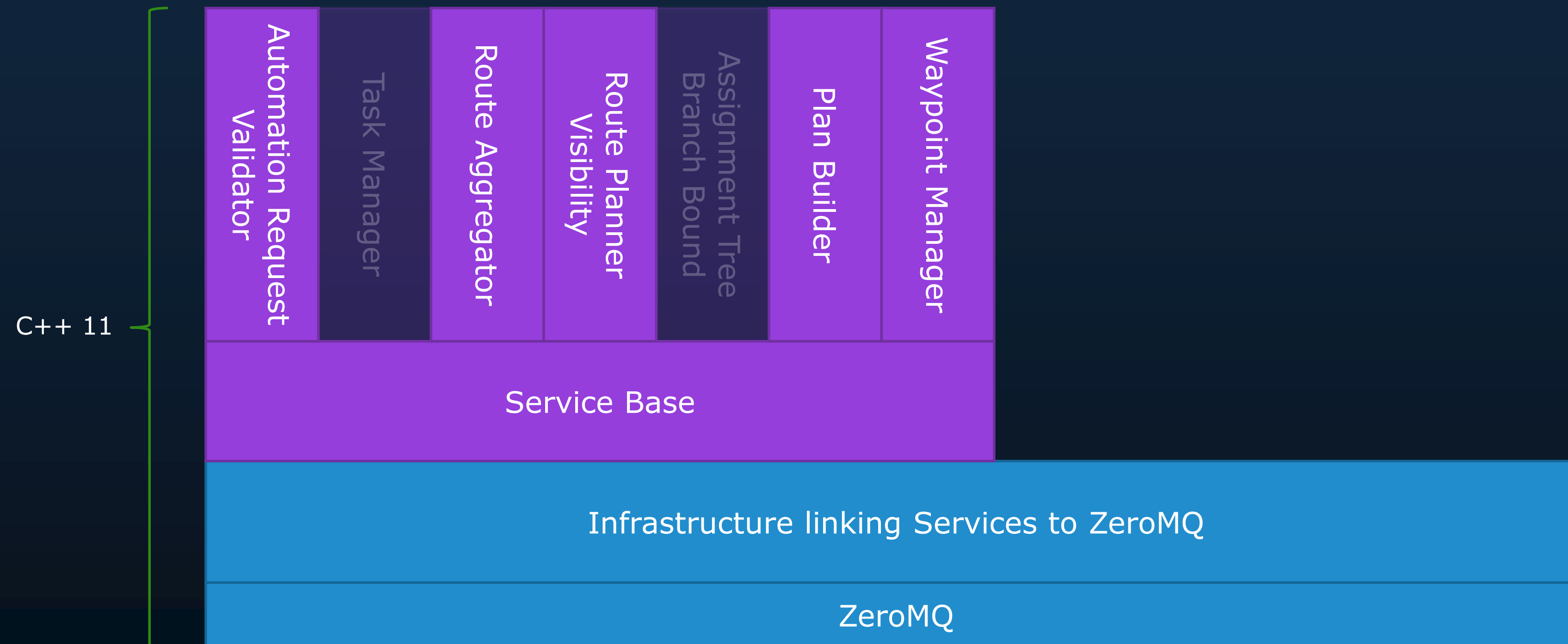
How Was OpenUxAS Originally Built?

Critical Code?

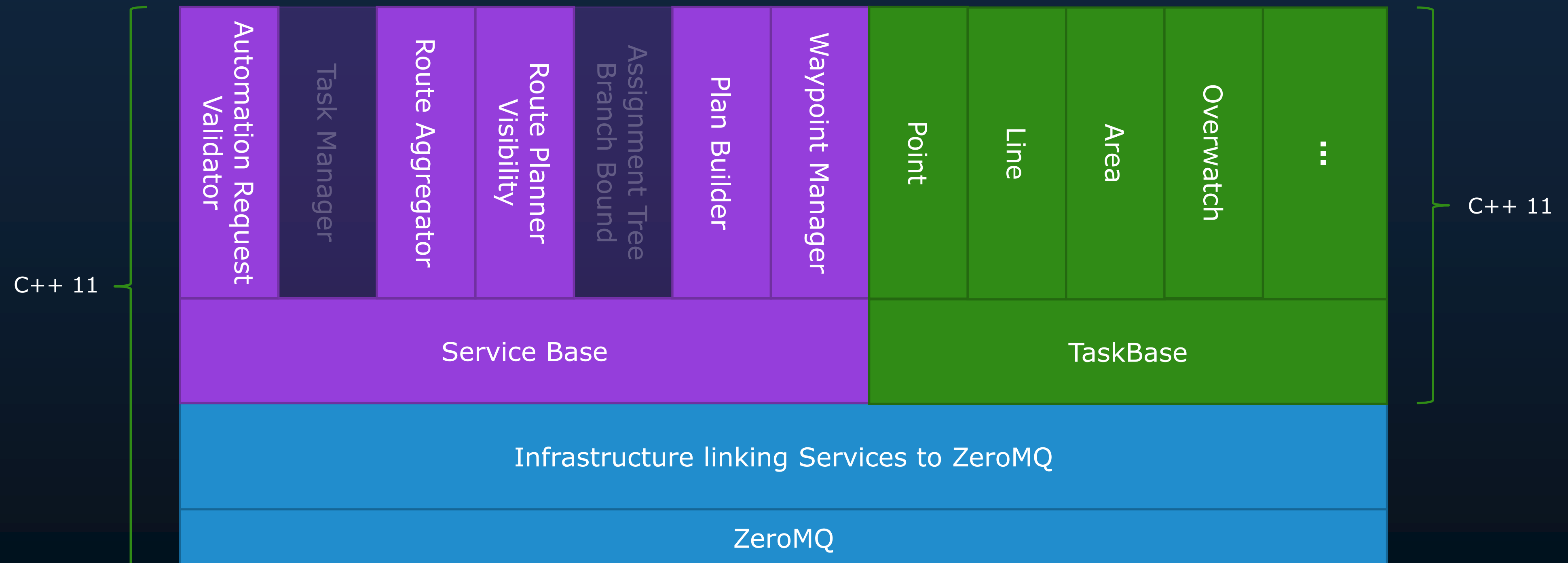
How Was OpenUxAS Originally Built?

Critical Code? In C++ 11?

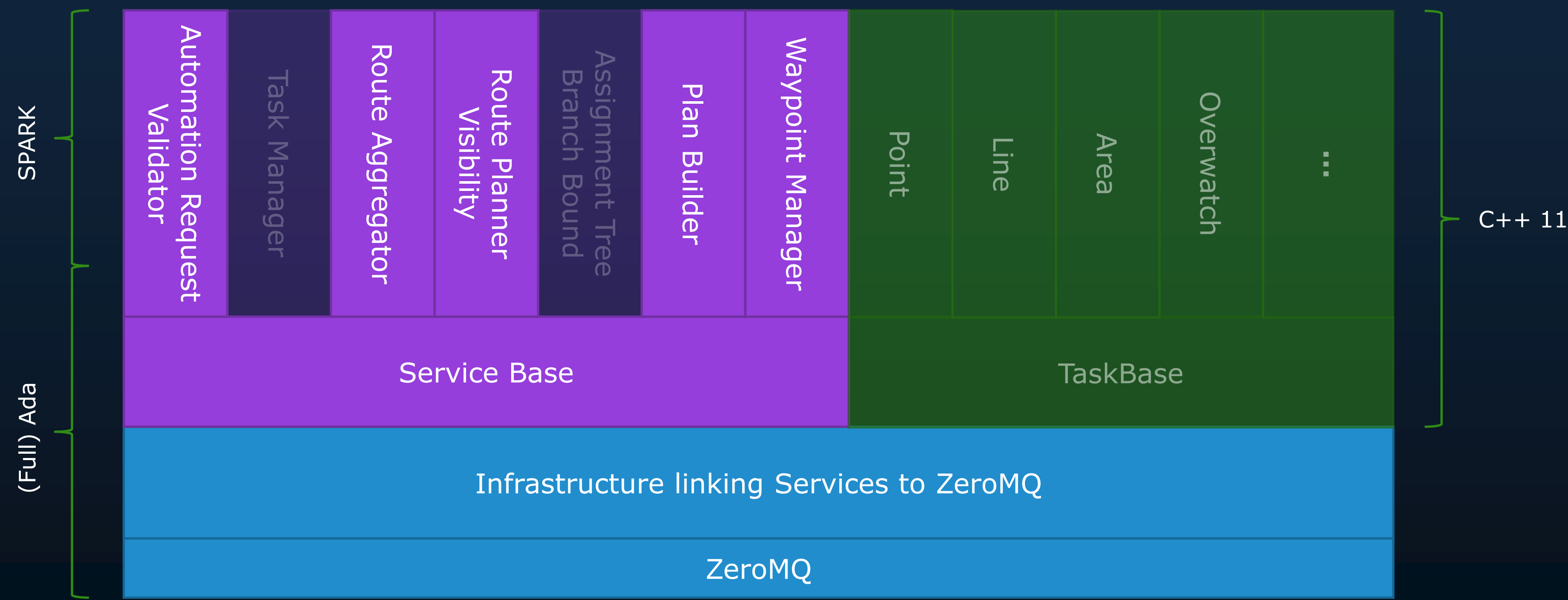
How Was OpenUxAS Originally Built?



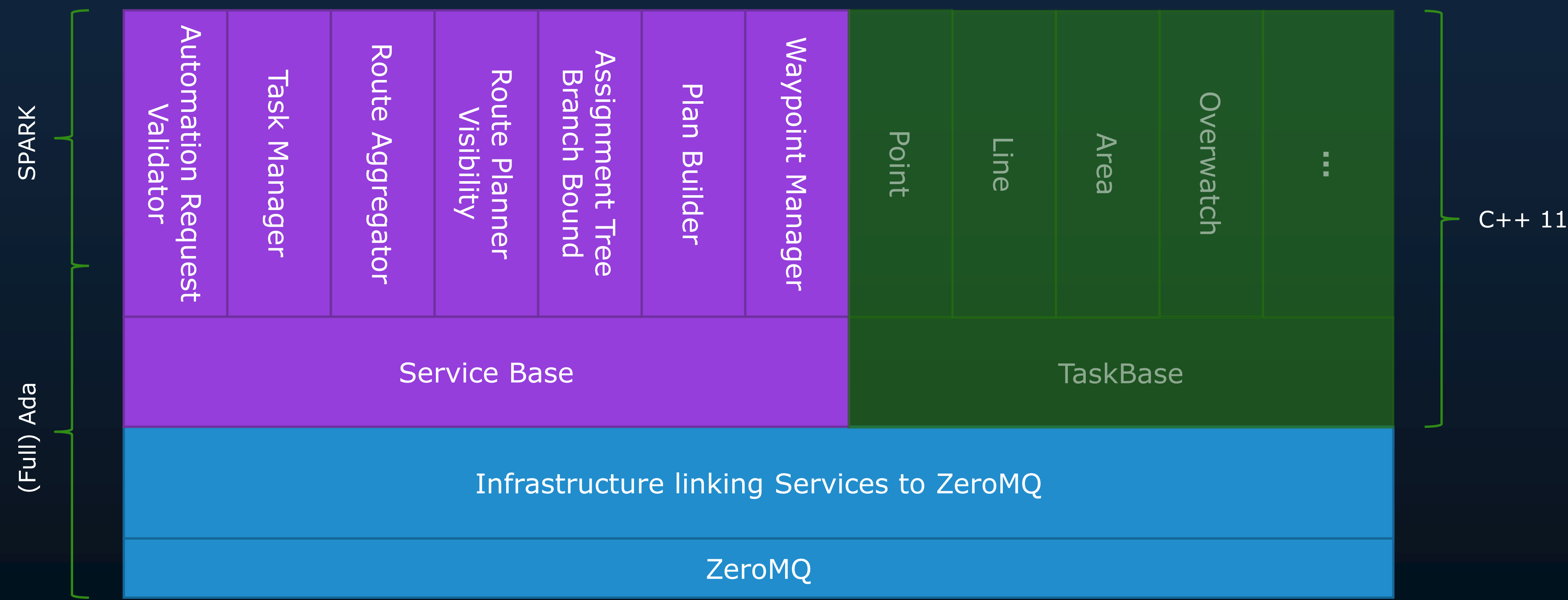
How Was OpenUxAS Originally Built?



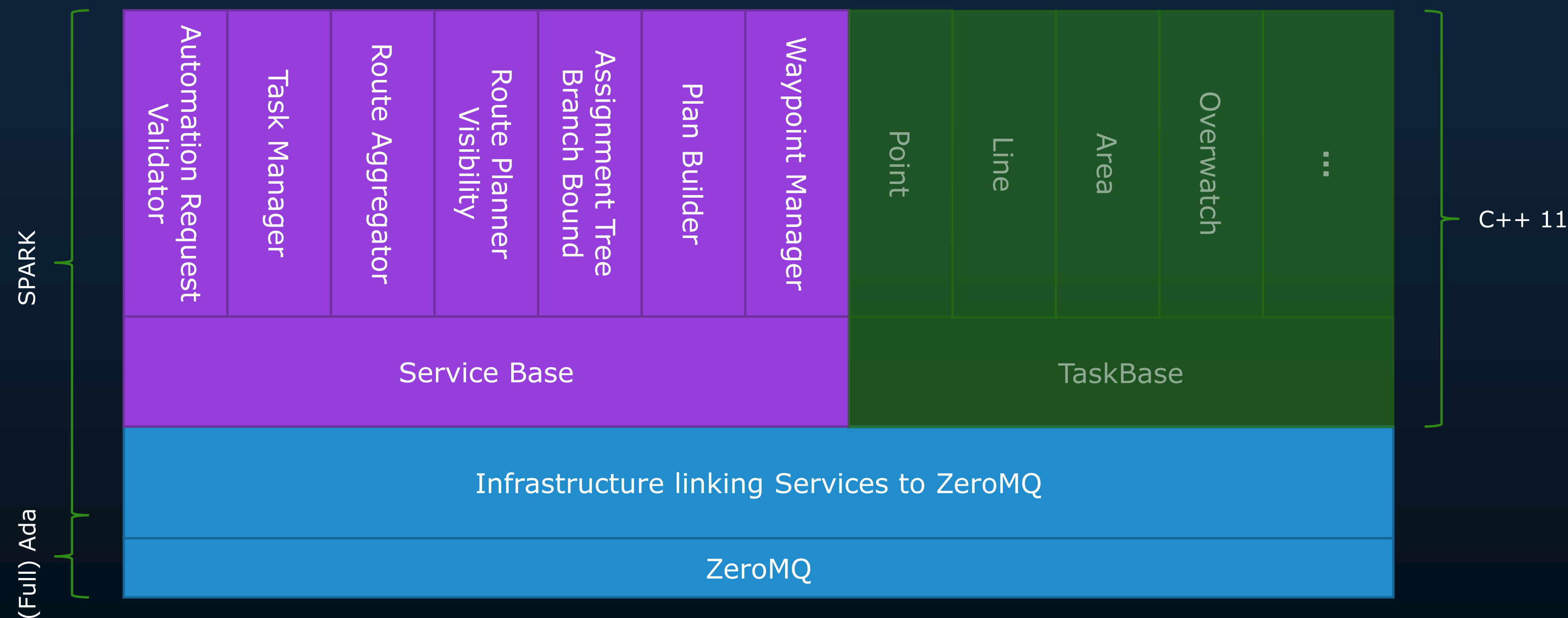
Goal: Critical Components



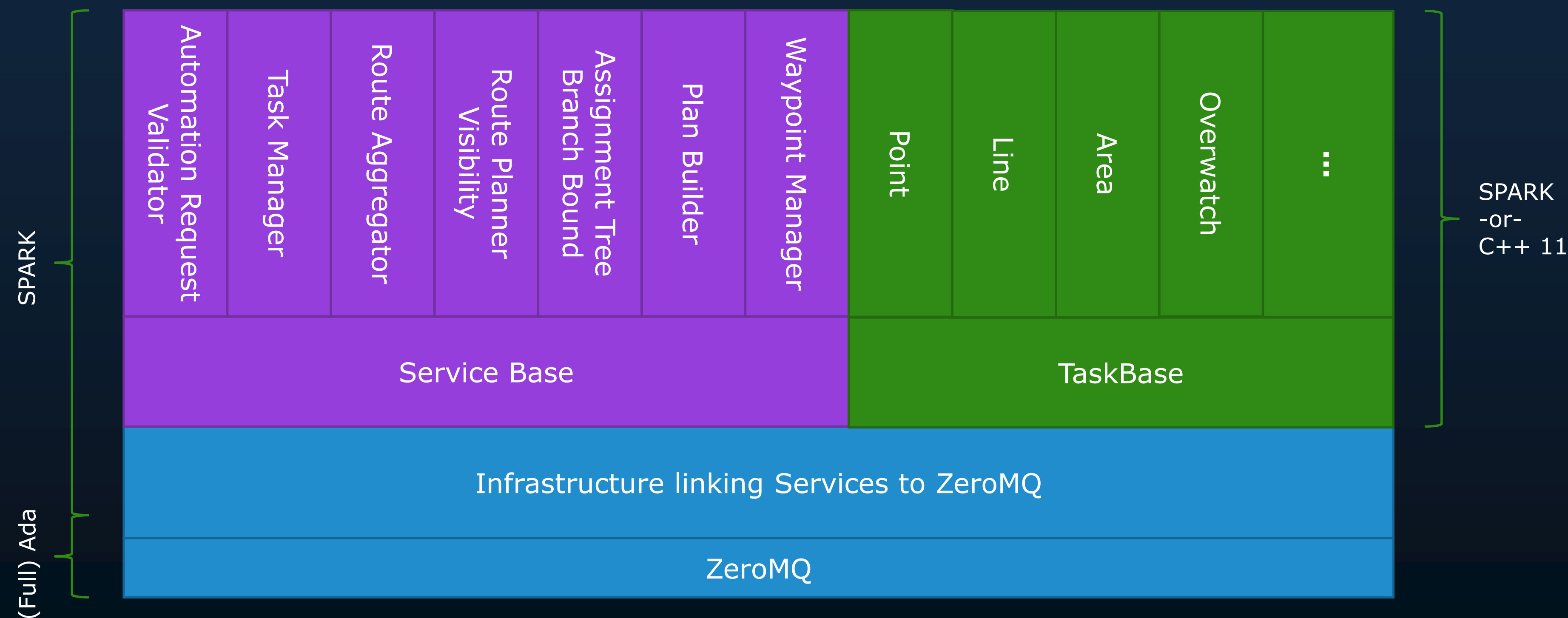
Goal: All Components



Goal: All Components in SPARK (and Ada)



Goal: All Components in SPARK (and Ada)



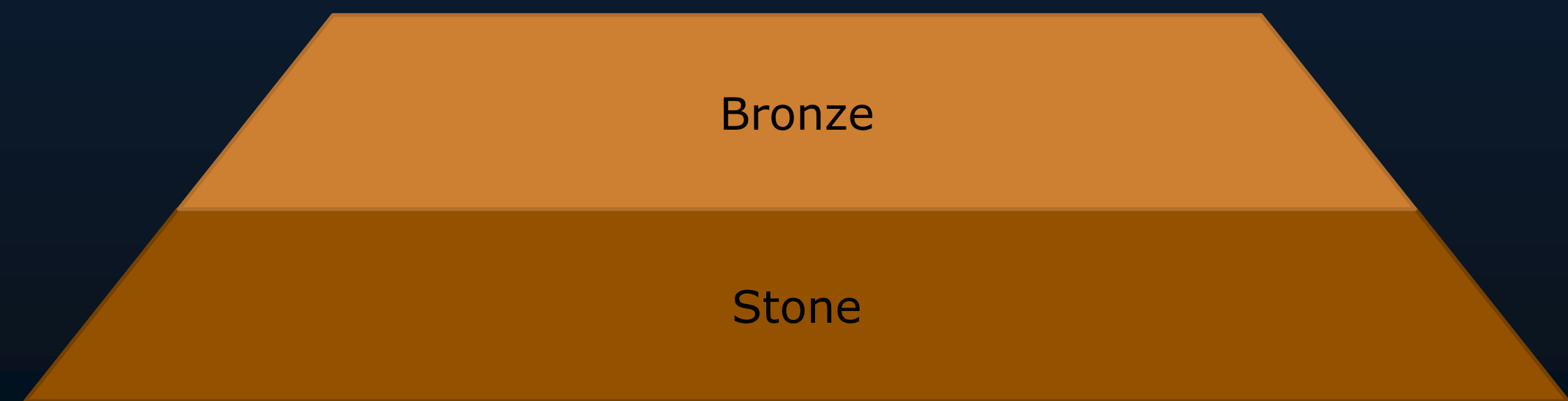
What We Get from SPARK



Stone

Use of the Language Subset: Benefit from
simply using SPARK

What We Get from SPARK



Flow Analysis: Not a significant part of our application

Use of the Language Subset: Benefit from simply using SPARK

What We Get from SPARK

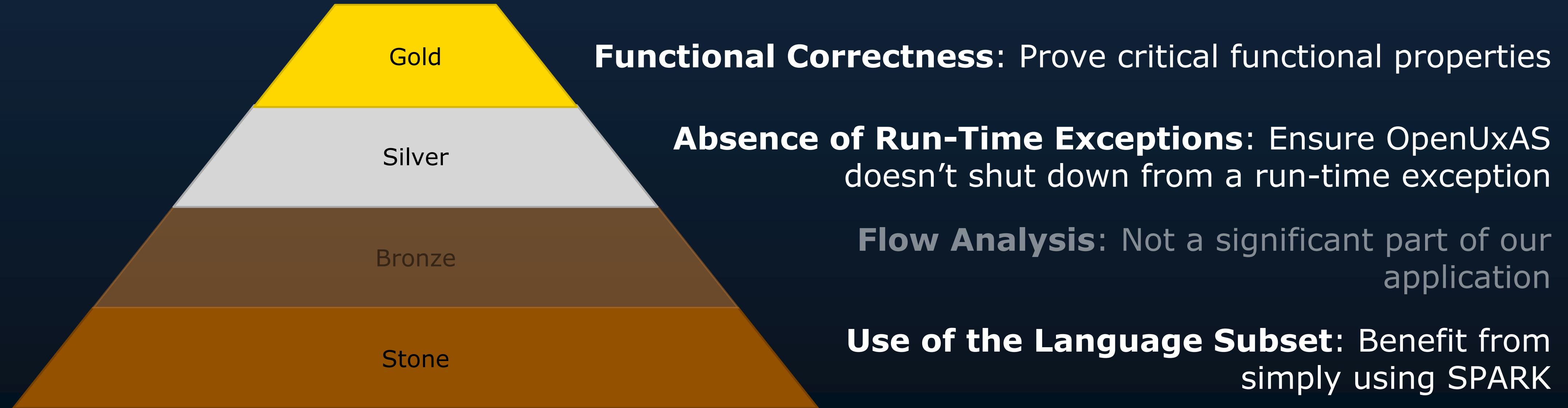


Absence of Run-Time Exceptions: Ensure OpenUxAS doesn't shut down from a run-time exception

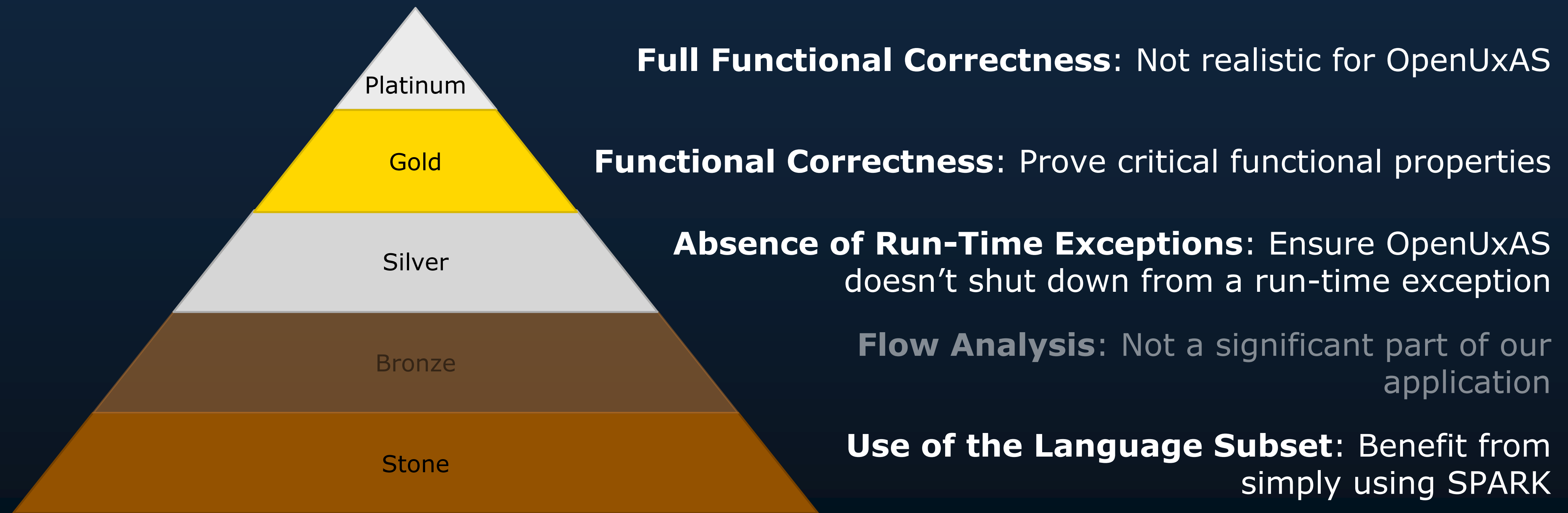
Flow Analysis: Not a significant part of our application

Use of the Language Subset: Benefit from simply using SPARK

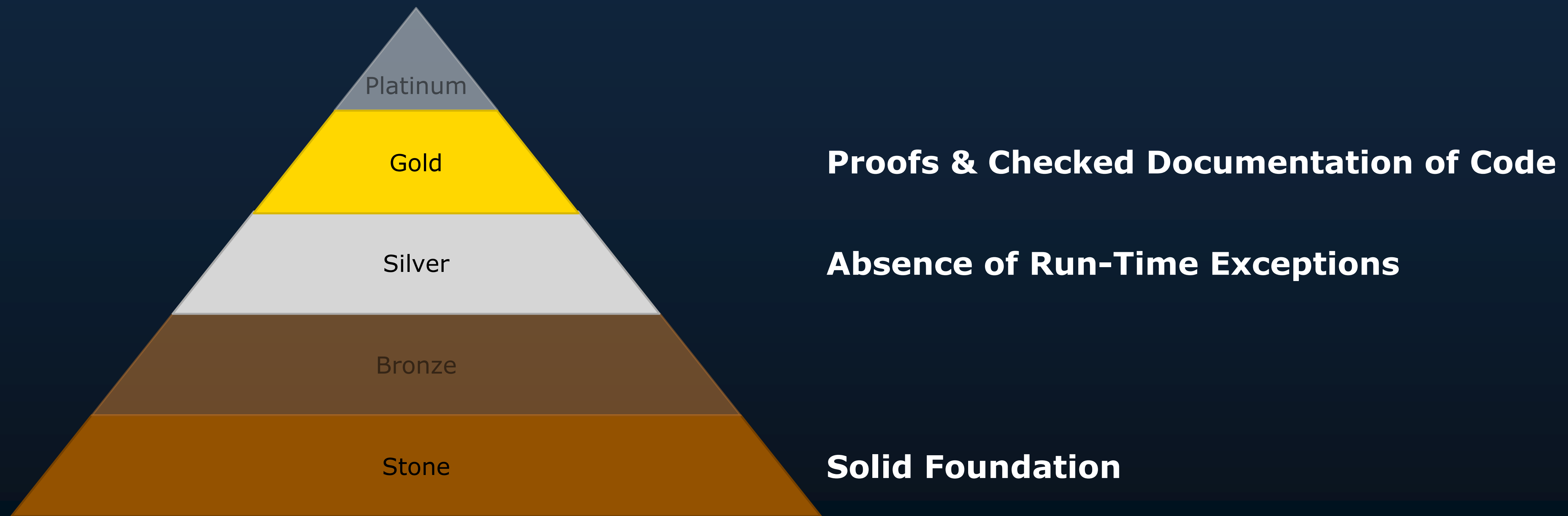
What We Get from SPARK



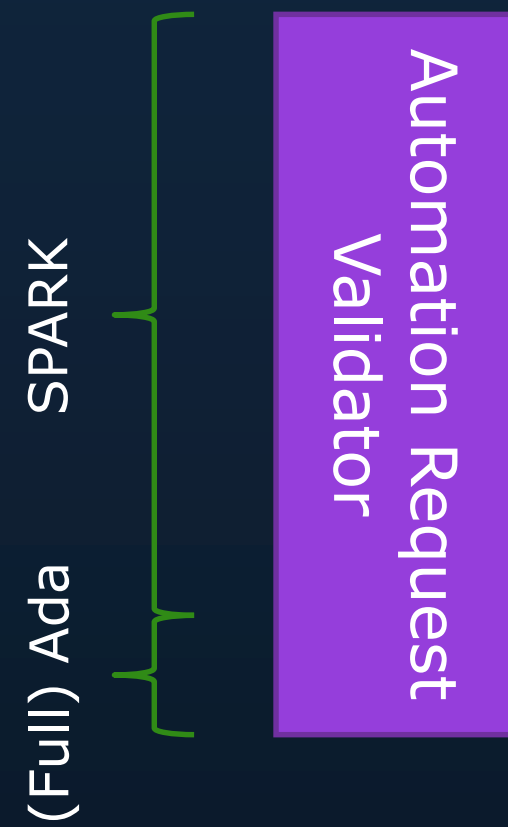
What We Get from SPARK



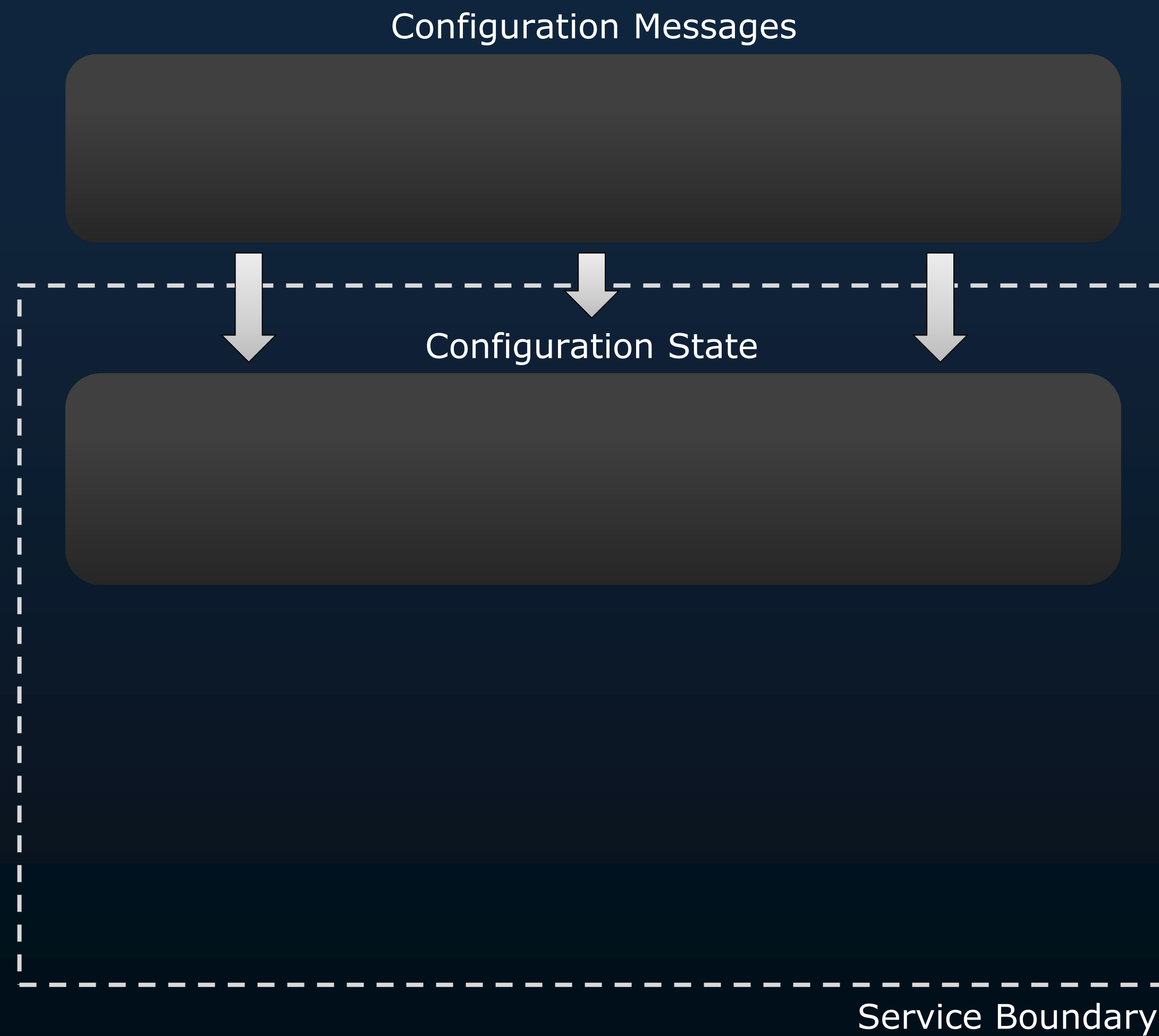
What We Get from SPARK



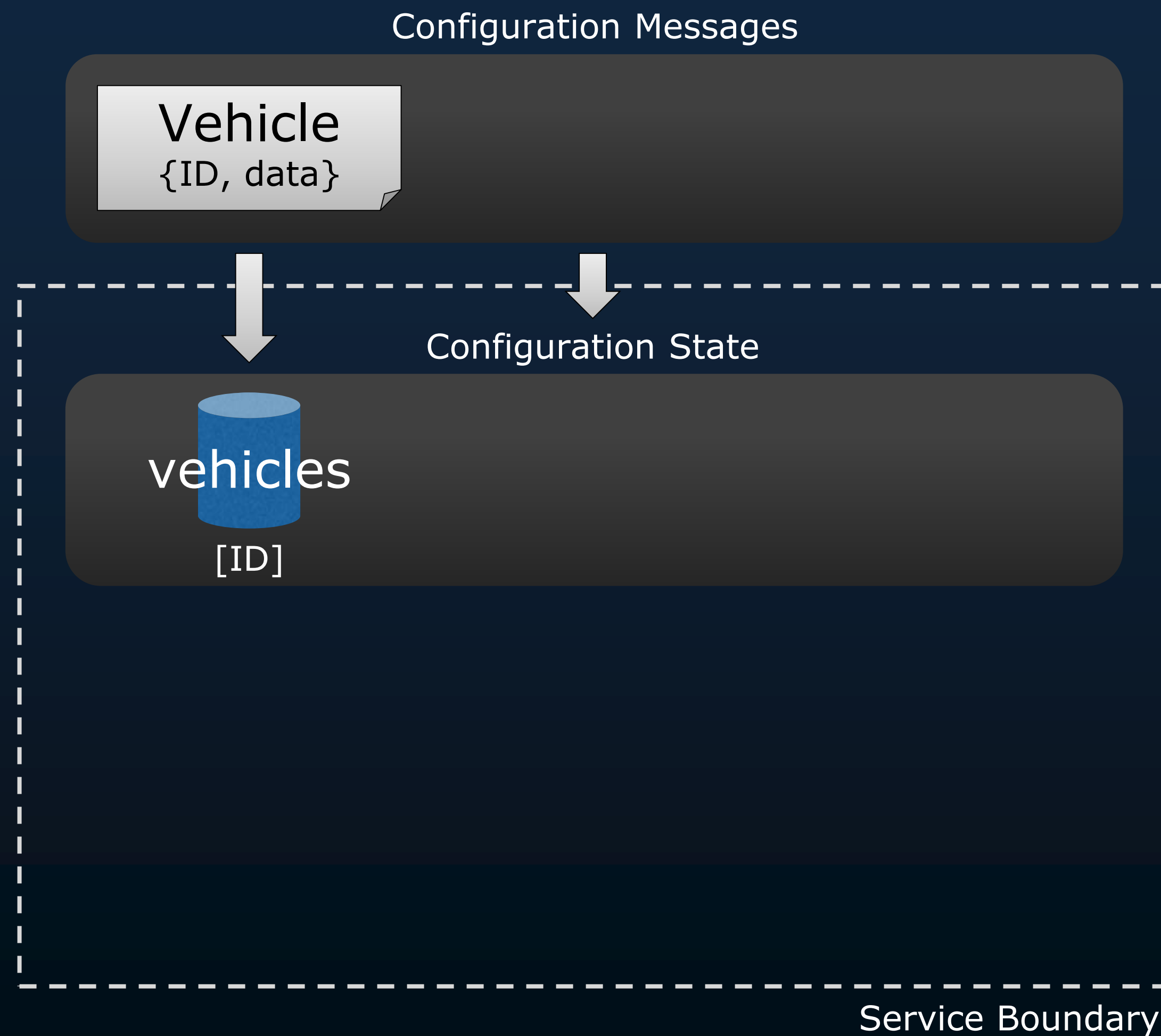
First Step



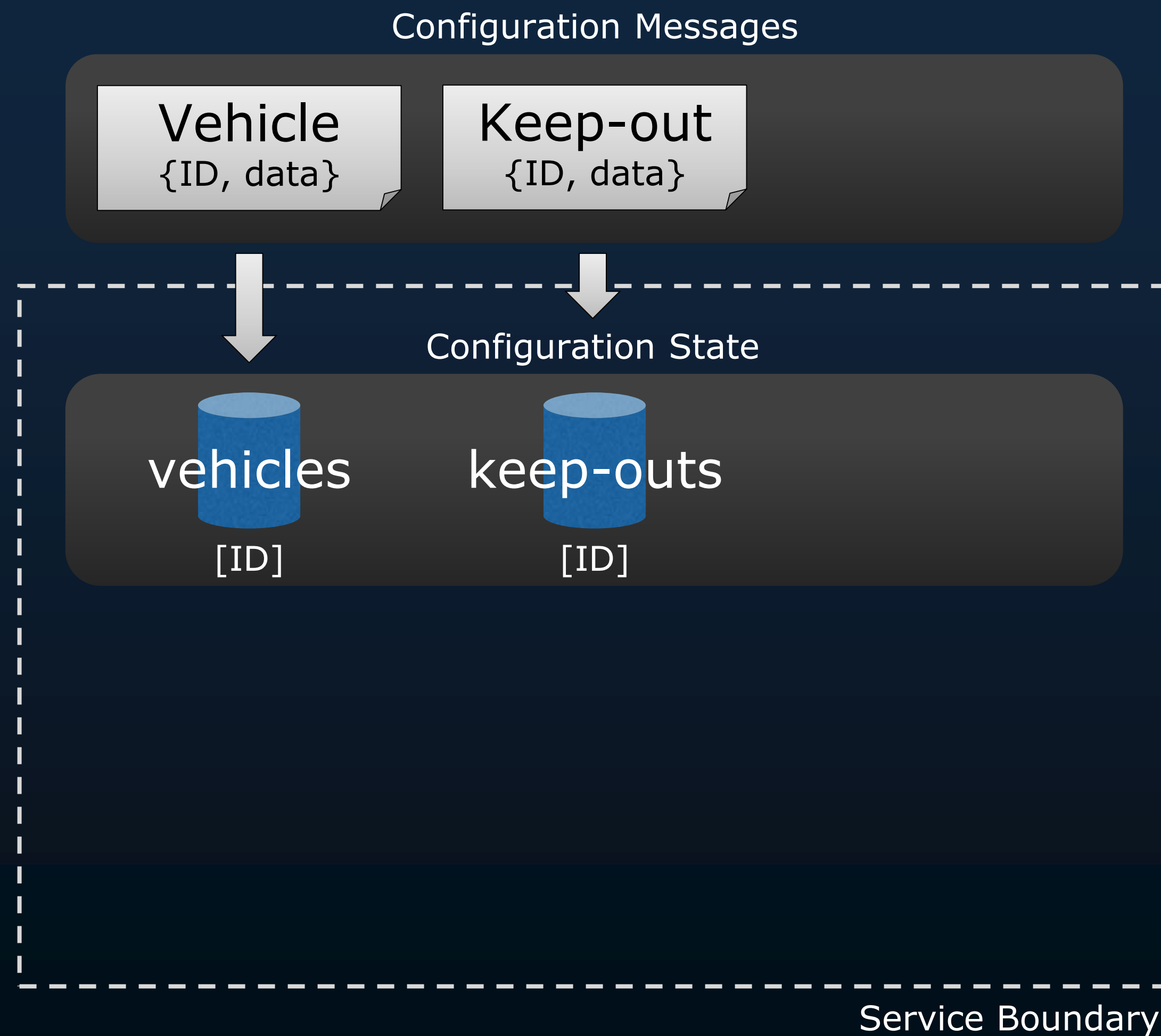
The Automation Request Validator



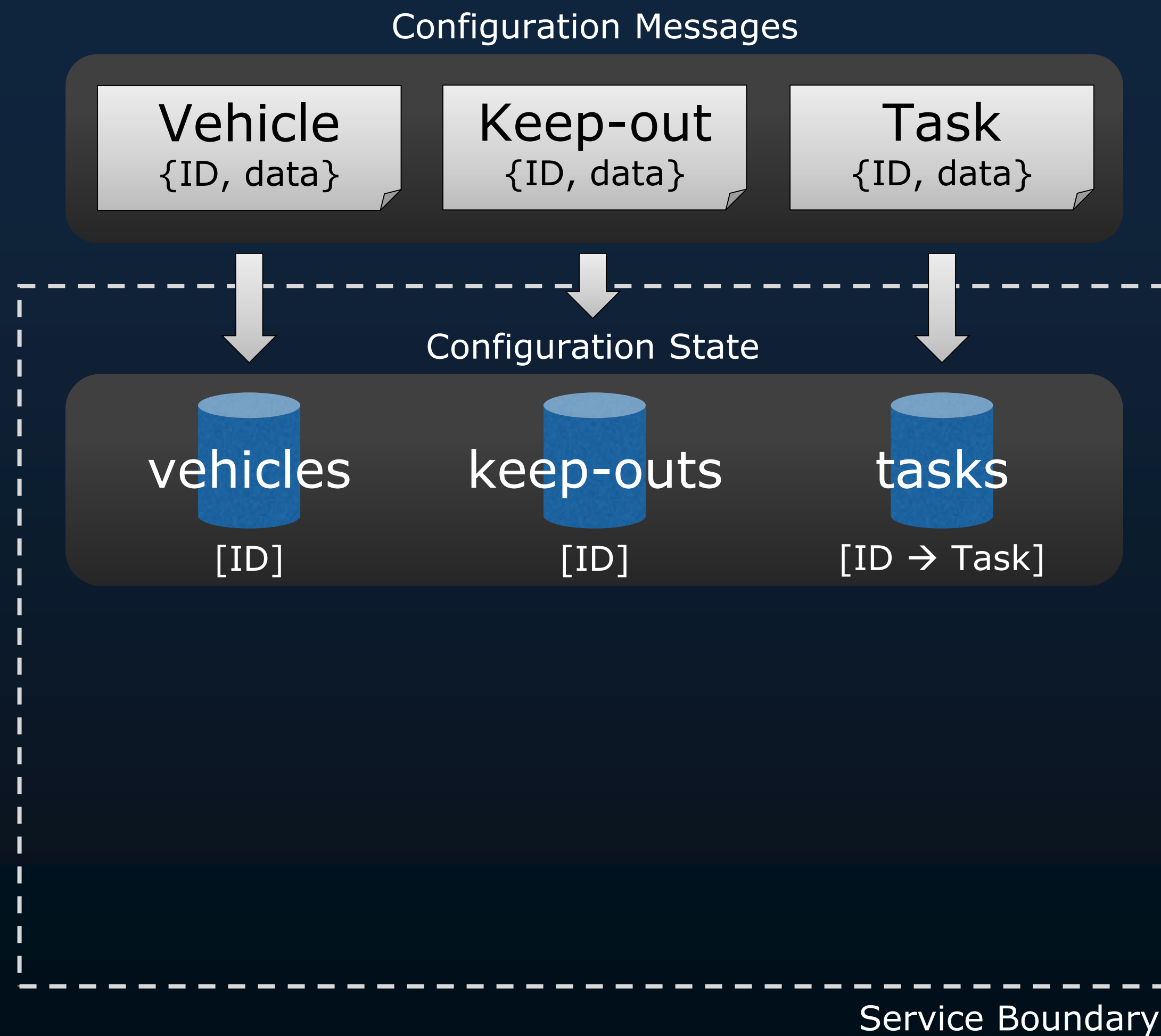
The Automation Request Validator



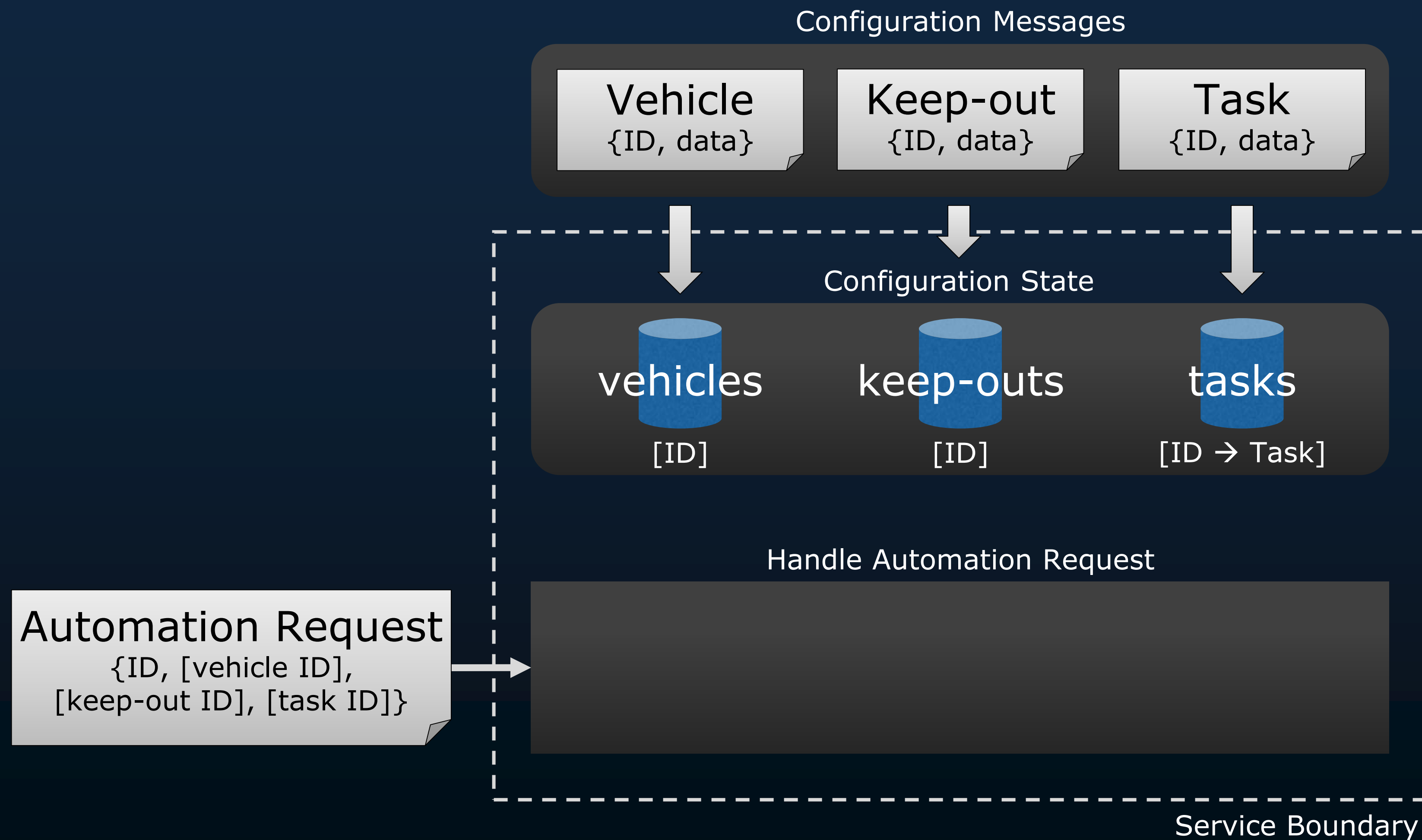
The Automation Request Validator



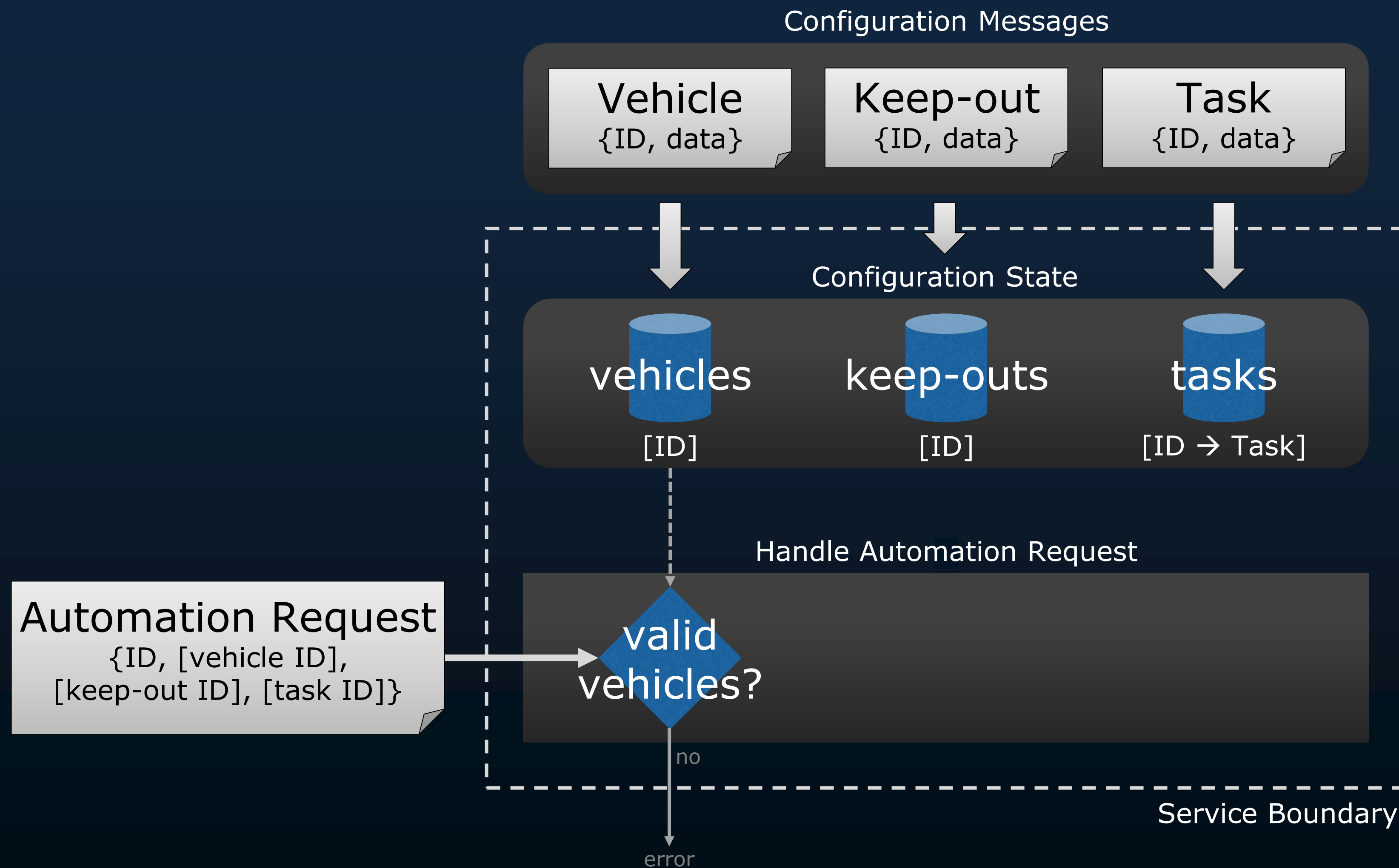
The Automation Request Validator



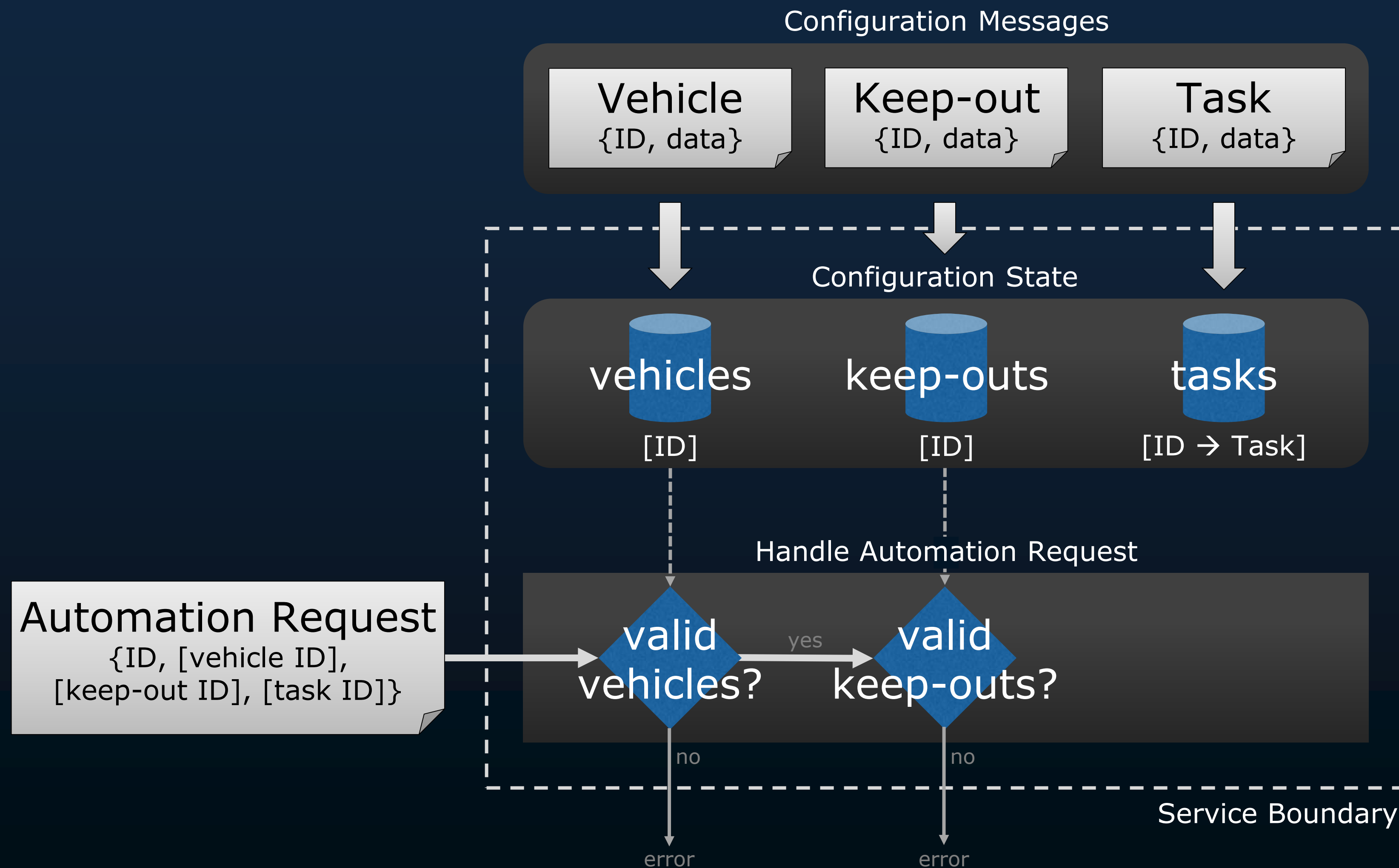
The Automation Request Validator



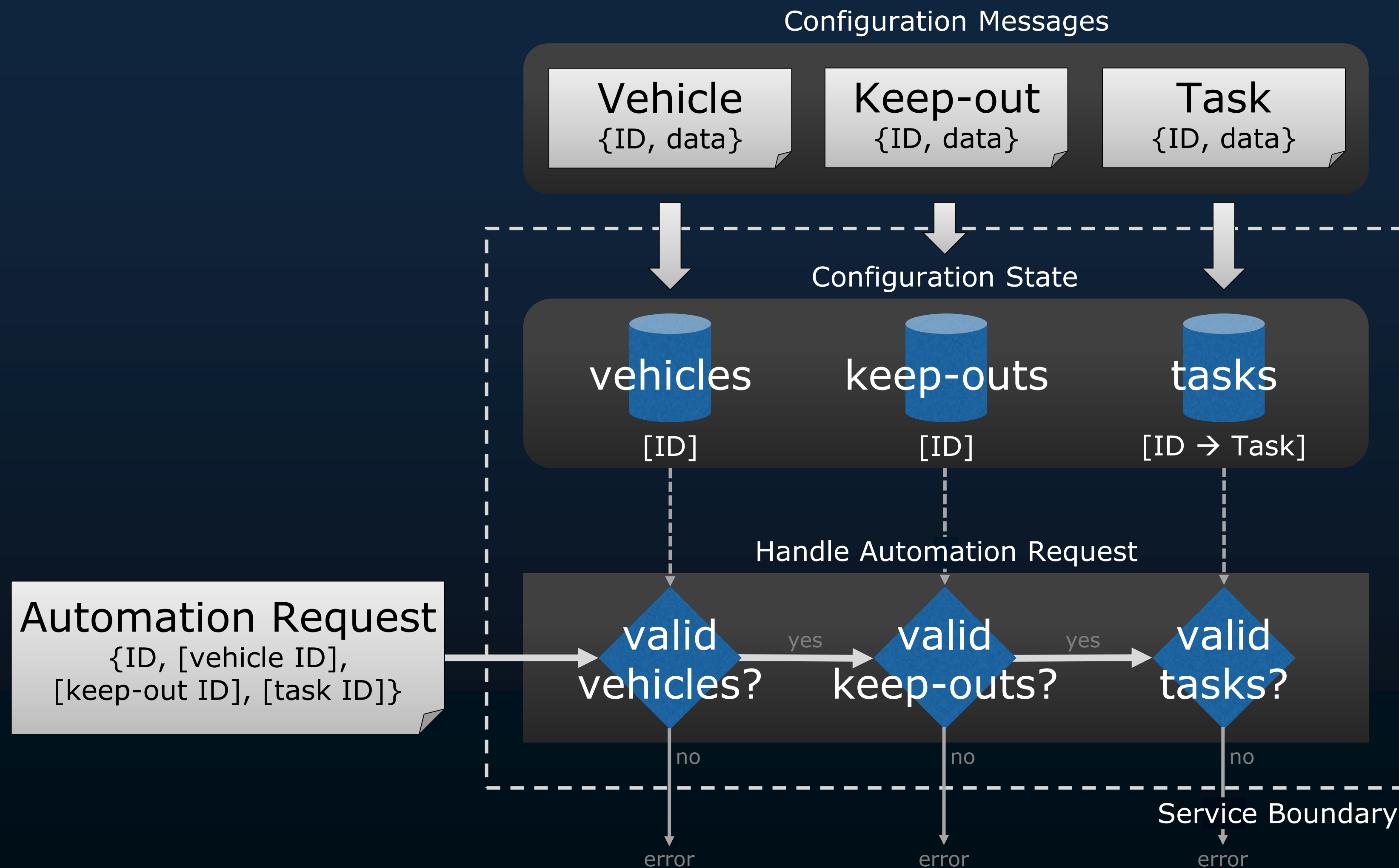
The Automation Request Validator



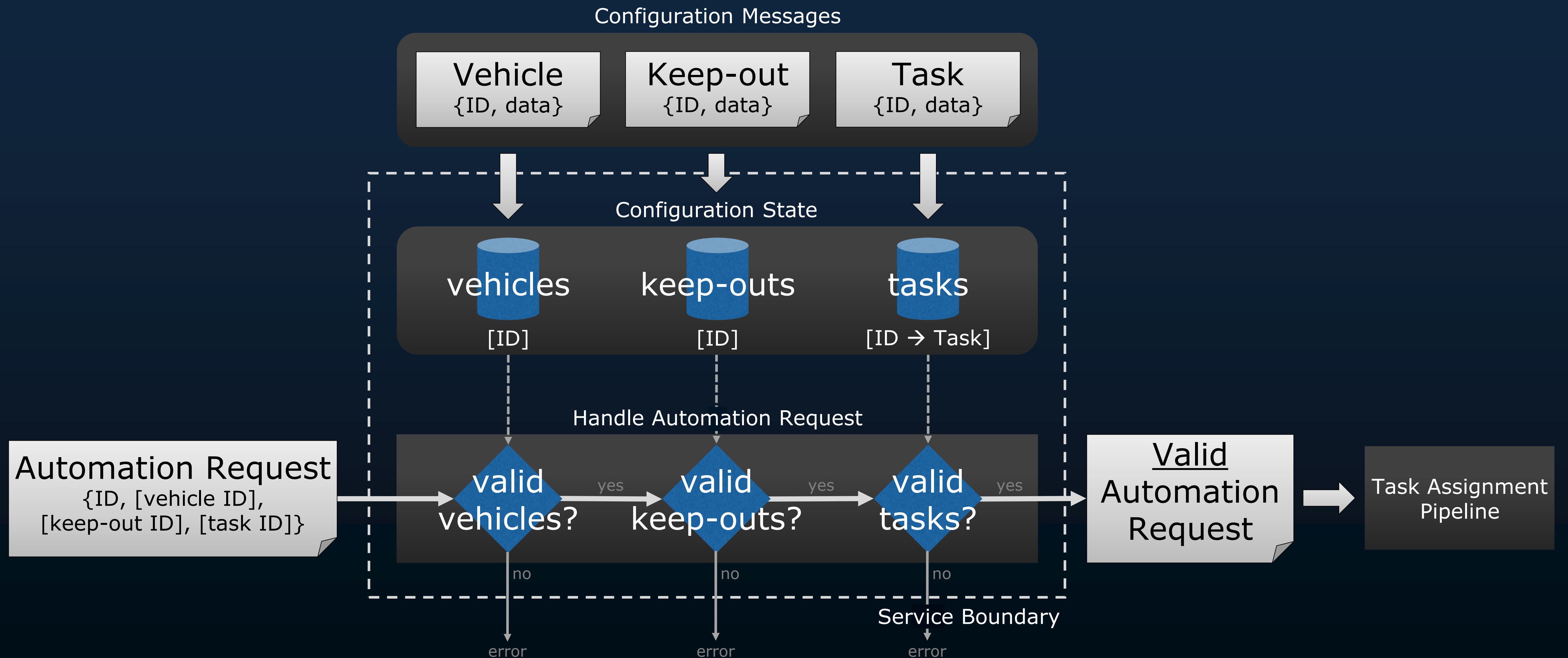
The Automation Request Validator



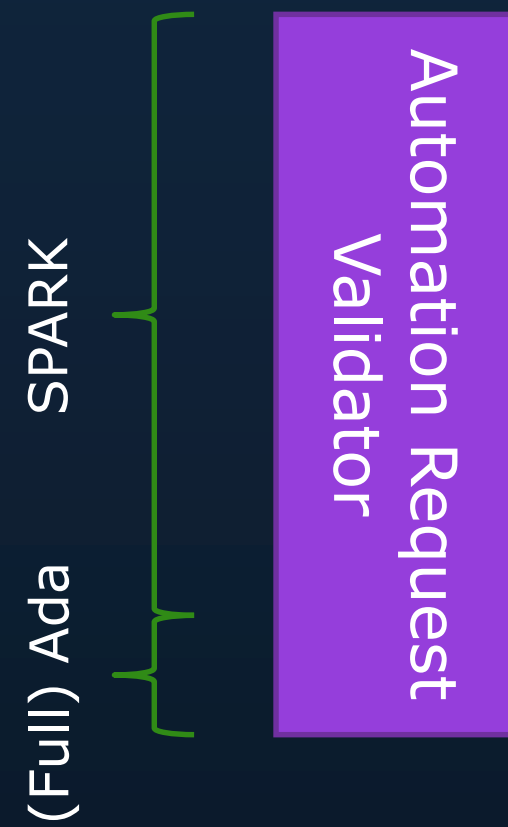
The Automation Request Validator



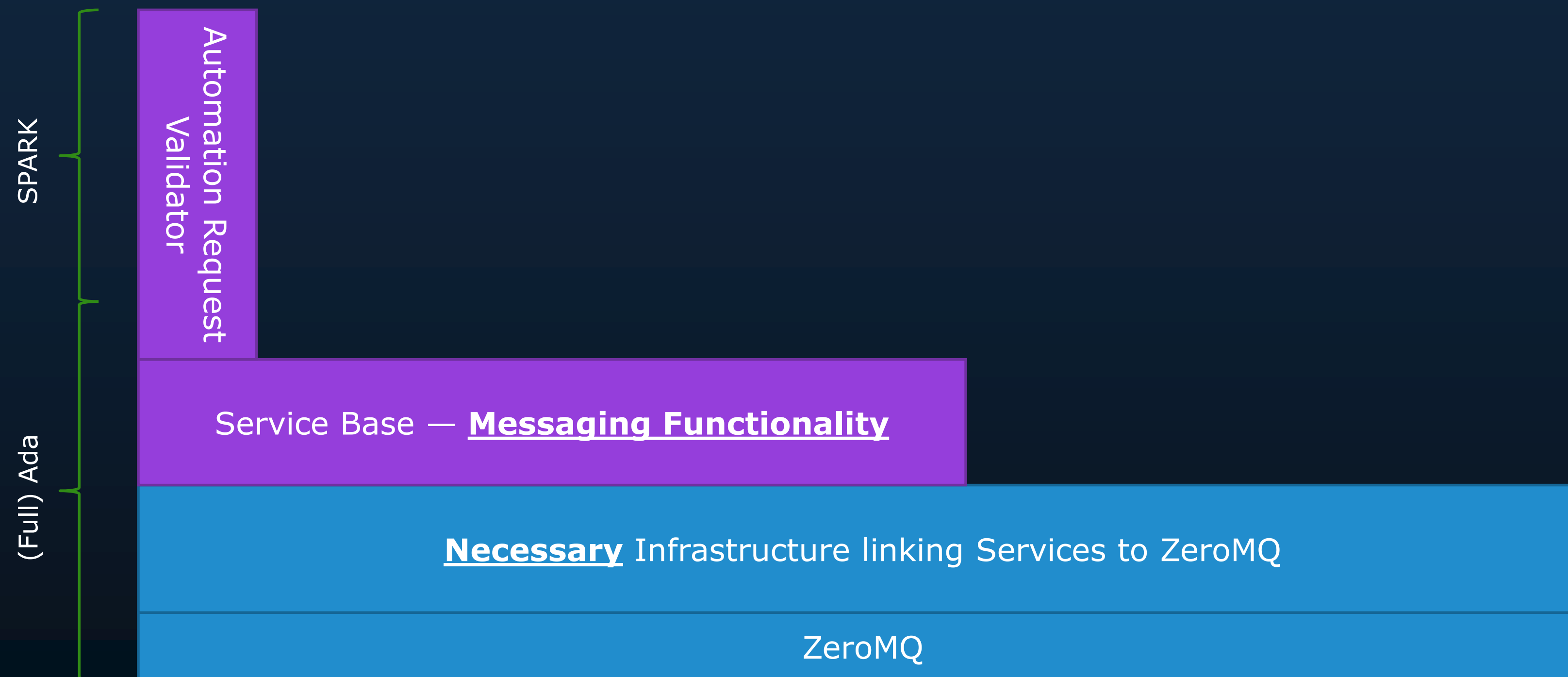
The Automation Request Validator



First Step



First Step



To Do This, We Had To

Update LMCP to automatically generate message types in Ada

Rebuild the service base in Ada and bind to ZeroMQ

Develop an Ada-SPARK interface for sharing message data

Formalize requirements for and prove correctness of the service

To Do This, We Had To

Update LMCP to automatically generate message types in Ada

Rebuild the service base in Ada and bind to ZeroMQ

Develop an Ada-SPARK interface for sharing message data

Formalize requirements for and prove correctness of the service

- **202 Ada Packages**
- **32,830 lines of (auto-generated) Ada**

To Do This, We Had To

Update LMCP to automatically generate message types in Ada

Rebuild the service base in Ada and bind to ZeroMQ

Develop an Ada-SPARK interface for sharing message data

Formalize requirements for and prove correctness of the service

- **30 Ada Packages**
- **7,015 lines of Ada**

To Do This, We Had To

Update LMCP to automatically generate message types in Ada

Rebuild the service base in Ada and bind to ZeroMQ

Develop an Ada-SPARK interface for sharing message data

Formalize requirements for and prove correctness of the service

- **11 SPARK Packages**
- **1,794 lines of SPARK**
- **10 predicates defining properties; 79 lines of property specification**

Additional Objective

Update LMCP to automatically generate message types in Ada

Rebuild the service base in Ada and bind to ZeroMQ

Develop an Ada-SPARK interface for sharing message data

Formalize requirements for and prove correctness of the service

Keep the Structure and Code as Close to the C++ as Possible

Facilitated Accuracy in the Translation

The Ada-SPARK Interface

Update LMCP to
automatically
generate
message types
in Ada

Rebuild the
service base in
Ada and bind to
ZeroMQ

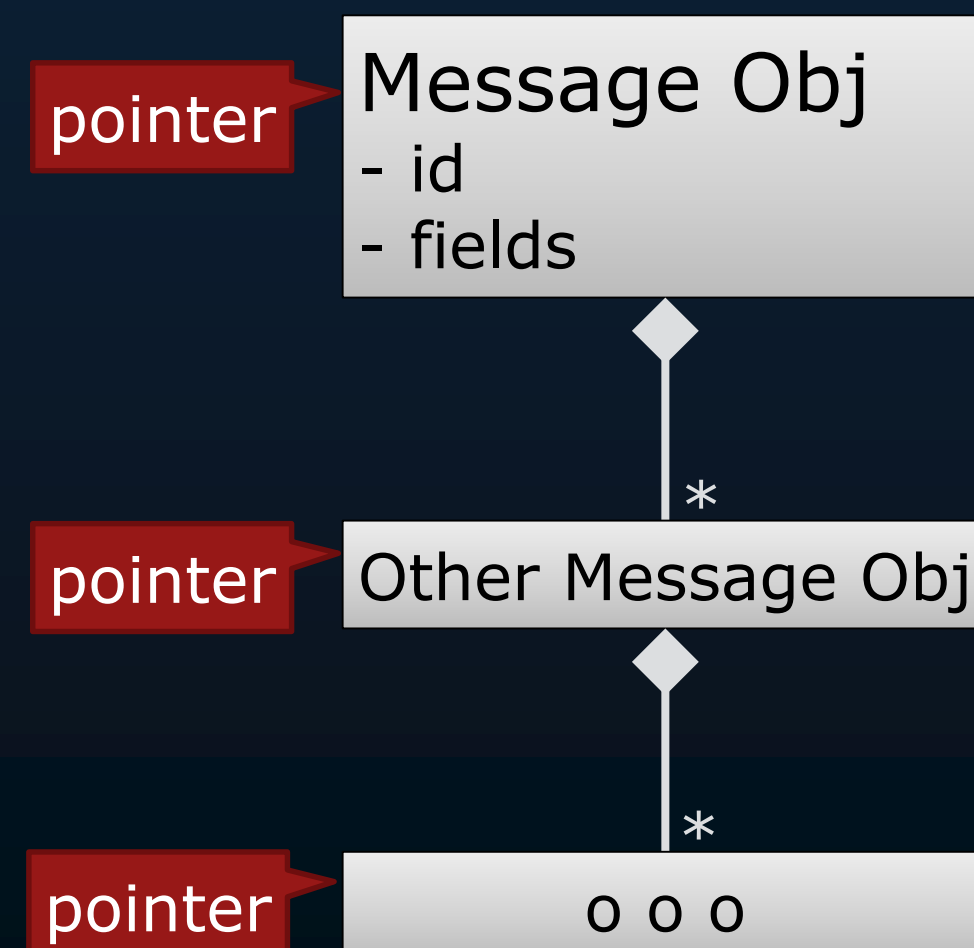
Develop an
Ada-SPARK
interface for
sharing
message data

Formalize
requirements
for and prove
correctness of
the service

Ada-SPARK Interface

C++

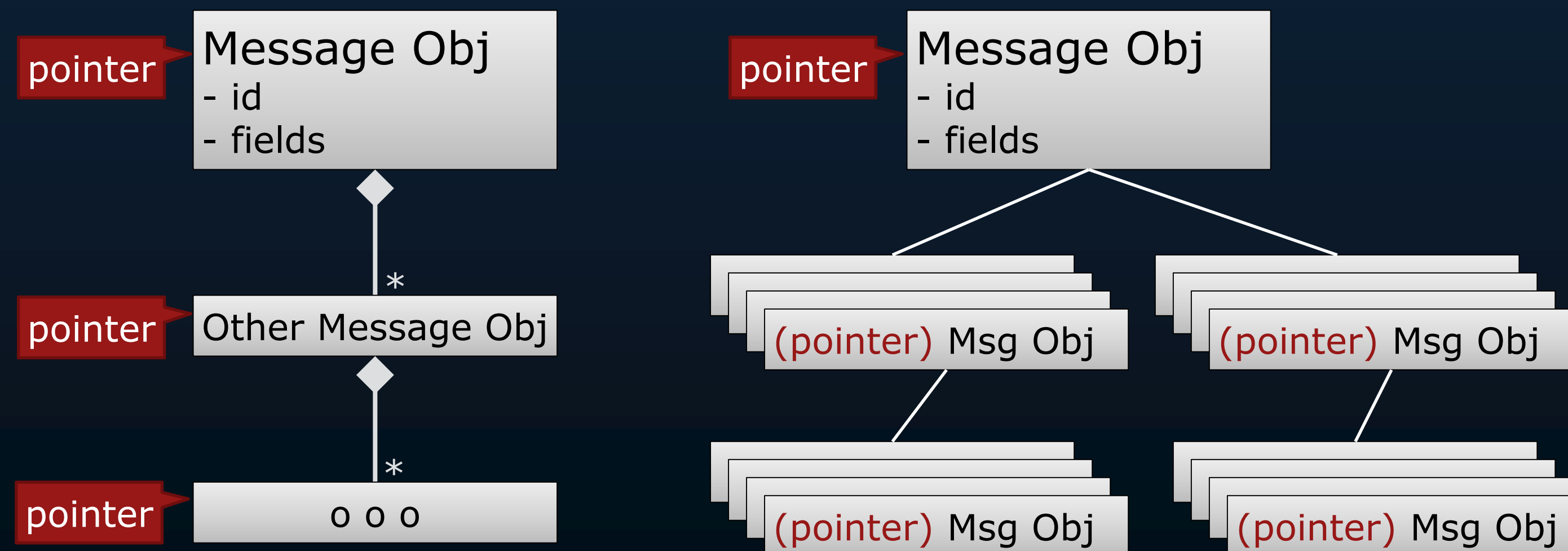
- Complex data model
- Heavy use of pointers
 - dynamic dispatch
 - reduce copying



Ada-SPARK Interface

C++

- Complex data model
- Heavy use of pointers
 - dynamic dispatch
 - reduce copying



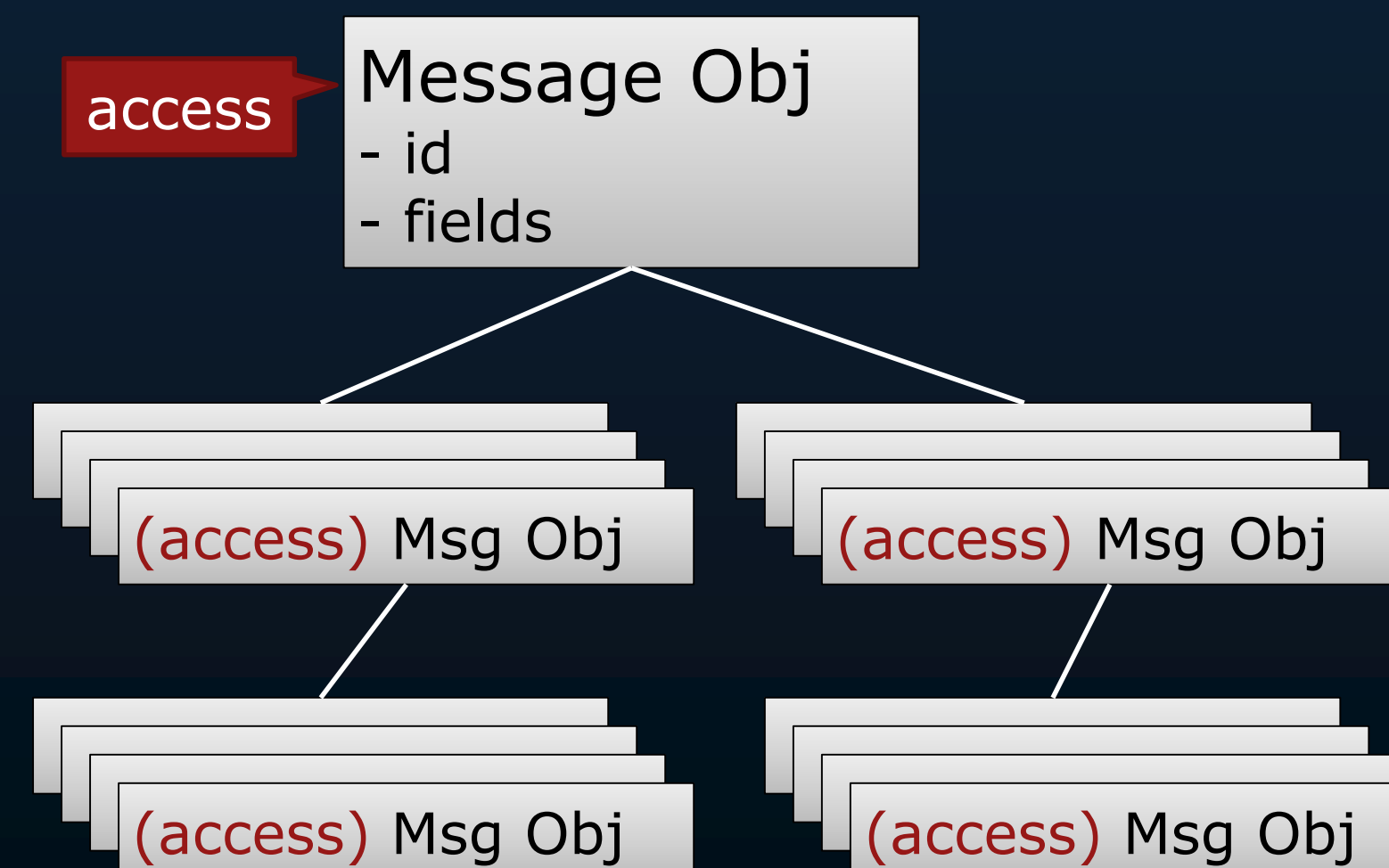
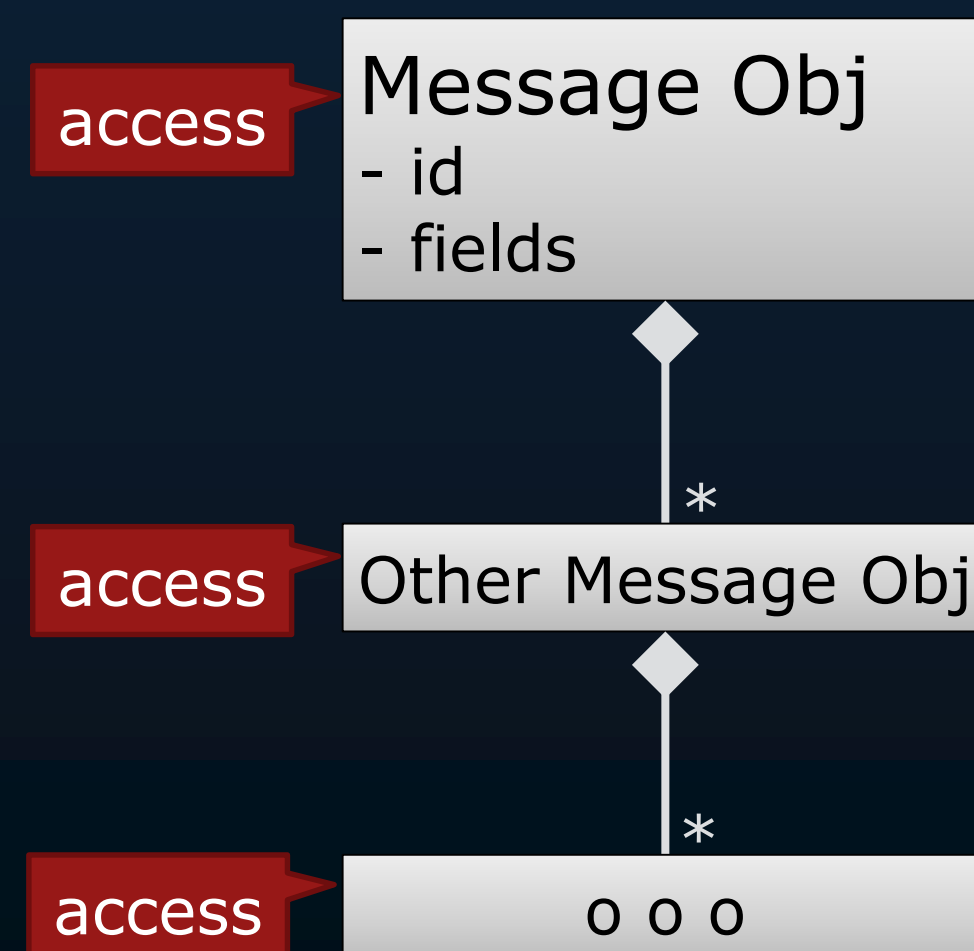
Ada-SPARK Interface

C++

- Complex data model
- Heavy use of pointers
 - dynamic dispatch
 - reduce copying

Ada

- Retain data model
- Same use of pointers
 - dynamic dispatch
 - reduce copying



Ada-SPARK Interface

C++

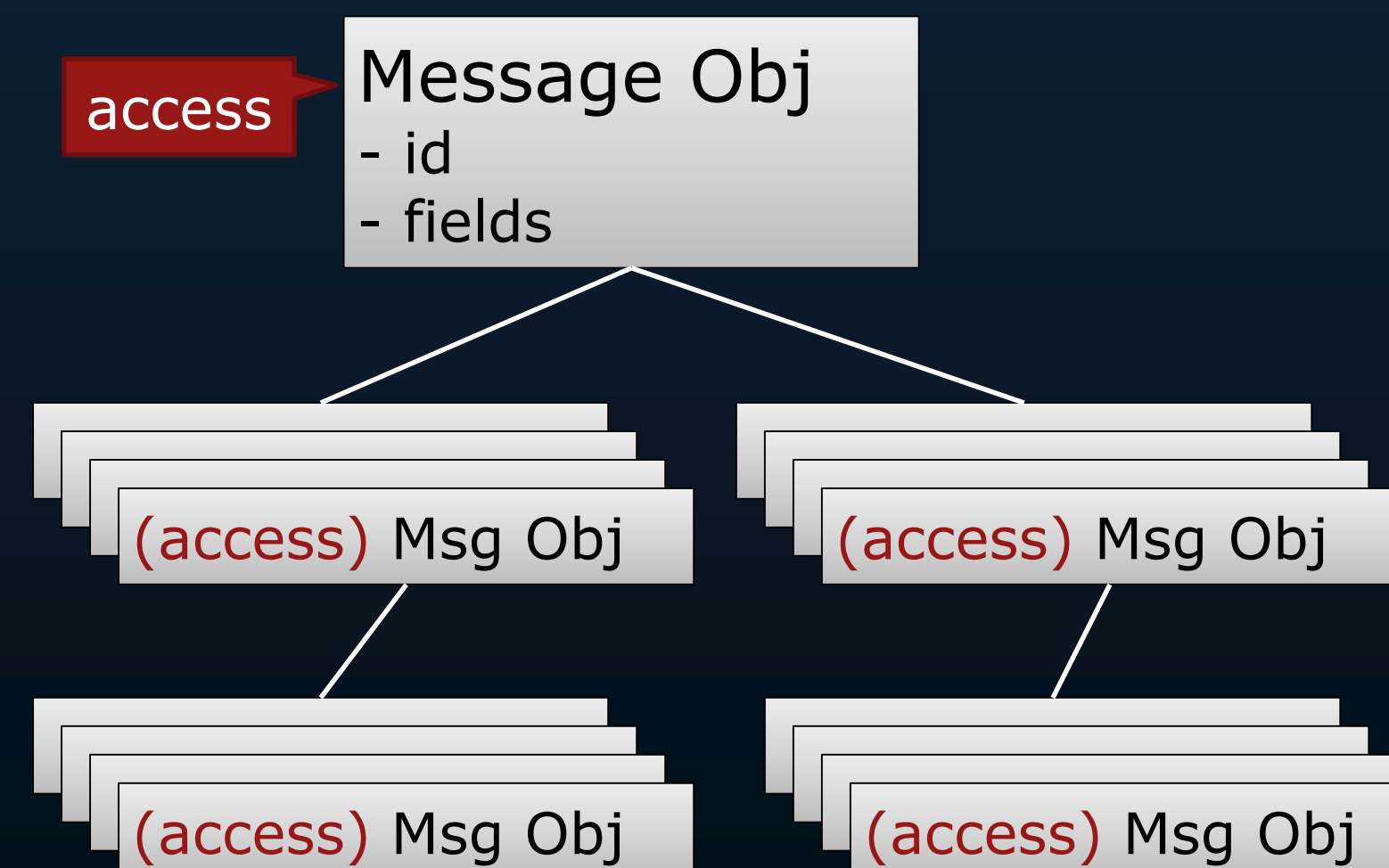
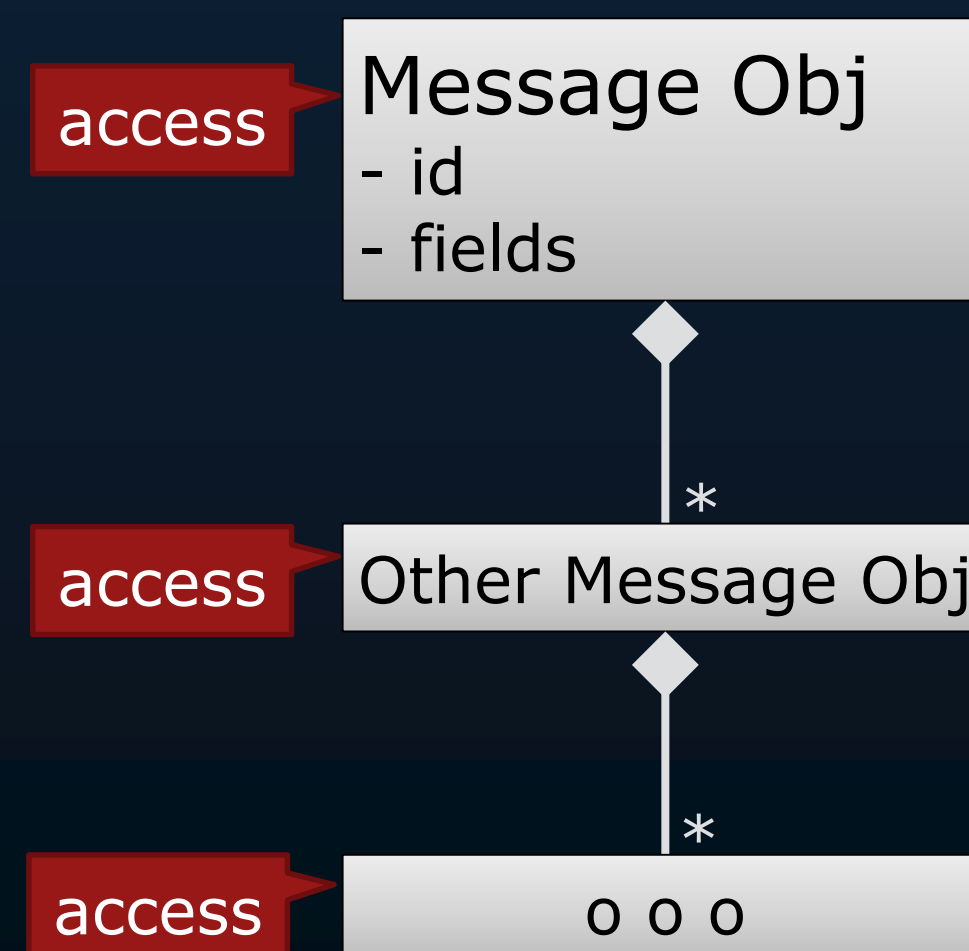
- Complex data model
- Heavy use of pointers
 - dynamic dispatch
 - reduce copying

Ada

- Retain data model
- Same use of pointers
 - dynamic dispatch
 - reduce copying

SPARK

- Restricted support for pointers
 - non-aliasing
 - not in tagged types



Ada-SPARK Interface

C++

- Complex data model
- Heavy use of pointers
 - dynamic dispatch
 - reduce copying

Ada

- Retain data model
- Same use of pointers
 - dynamic dispatch
 - reduce copying

SPARK

- Restricted support for pointers
 - non-aliasing
 - not in tagged types



Ada-SPARK Interface

C++

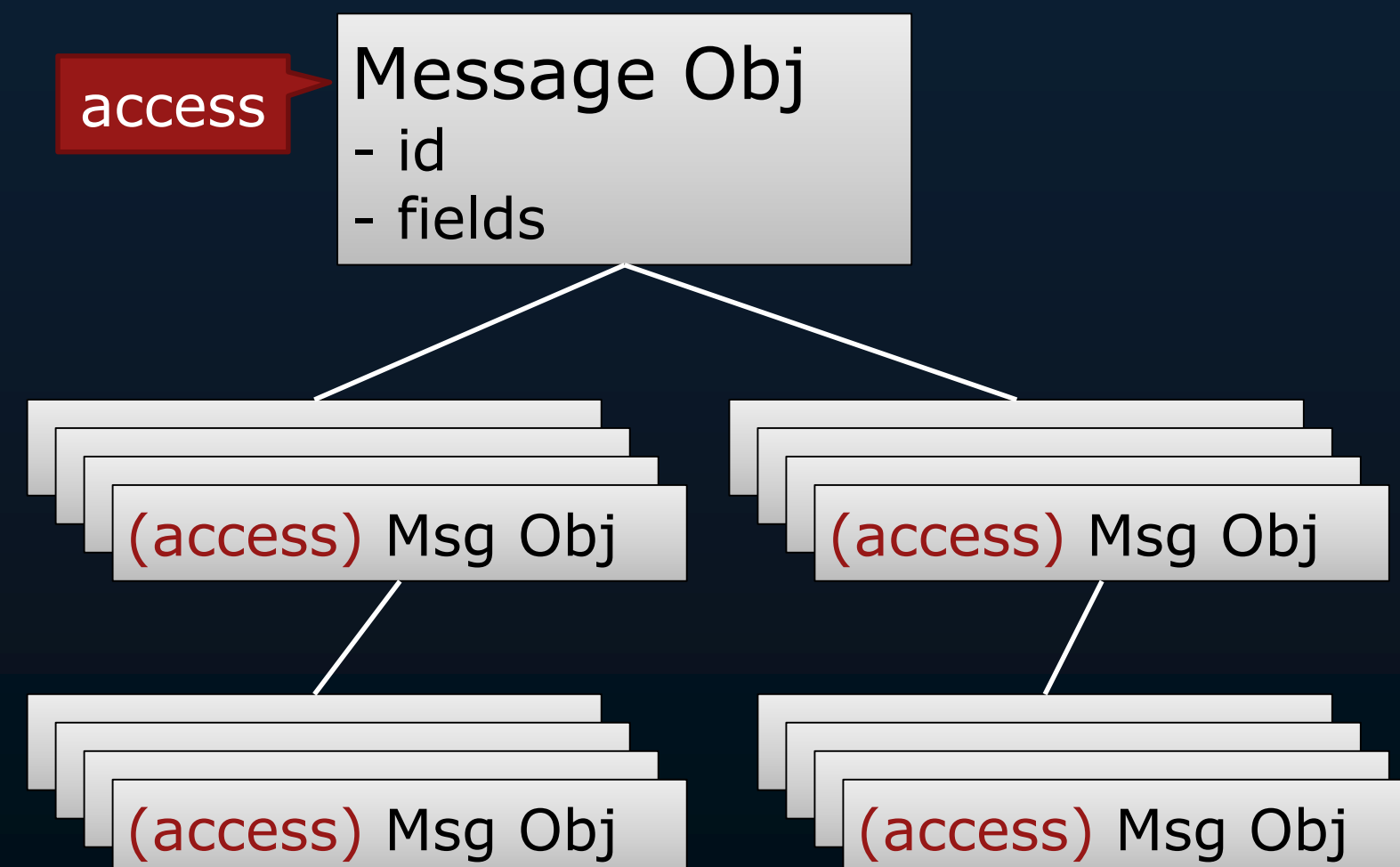
- Complex data model
- Heavy use of pointers
 - dynamic dispatch
 - reduce copying

Ada

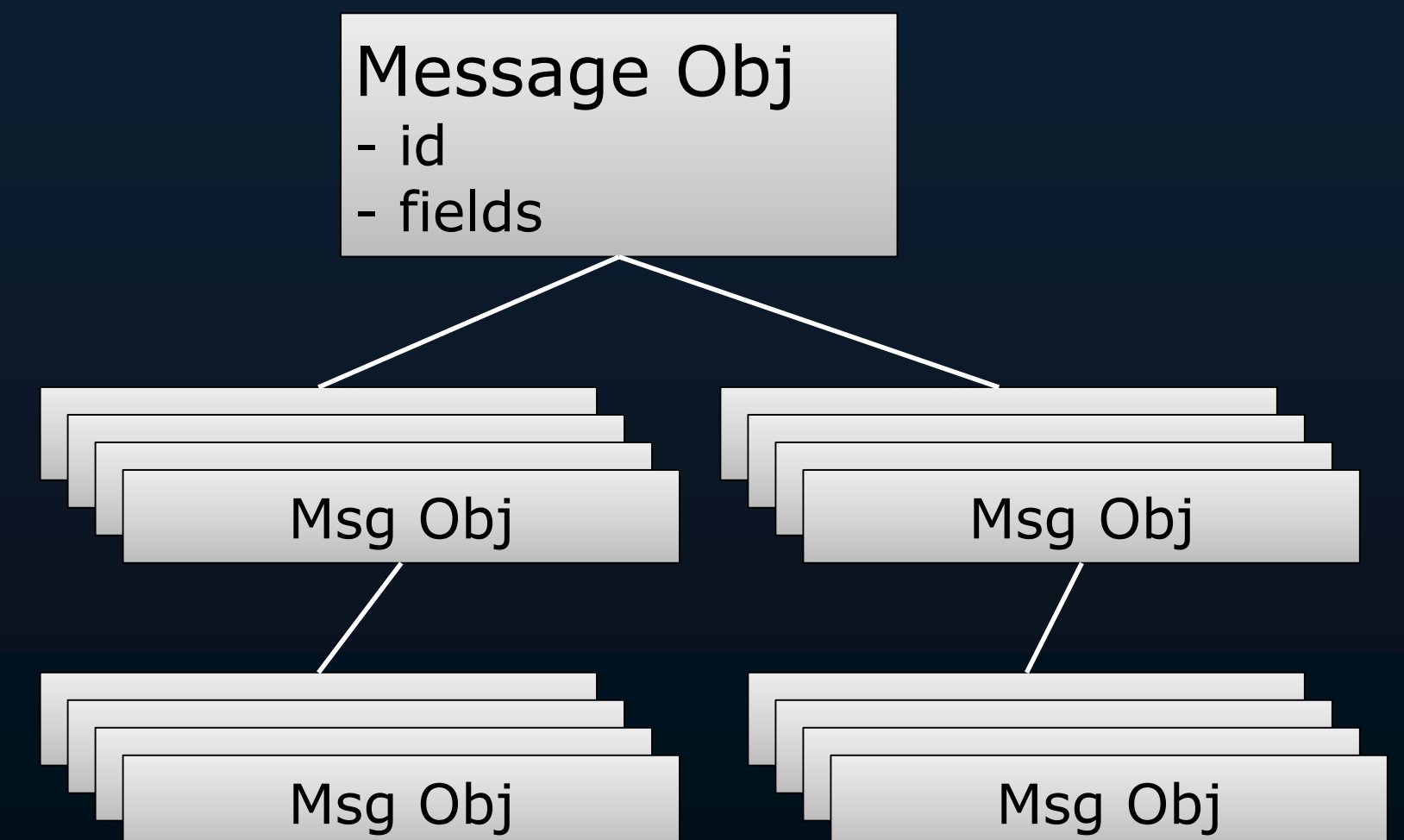
- Retain data model
- Same use of pointers
 - dynamic dispatch
 - reduce copying

SPARK

- Restricted support for pointers
 - non-aliasing
 - not in tagged types



! →
deep copy &
replace pointers



Ada-SPARK Interface

C++

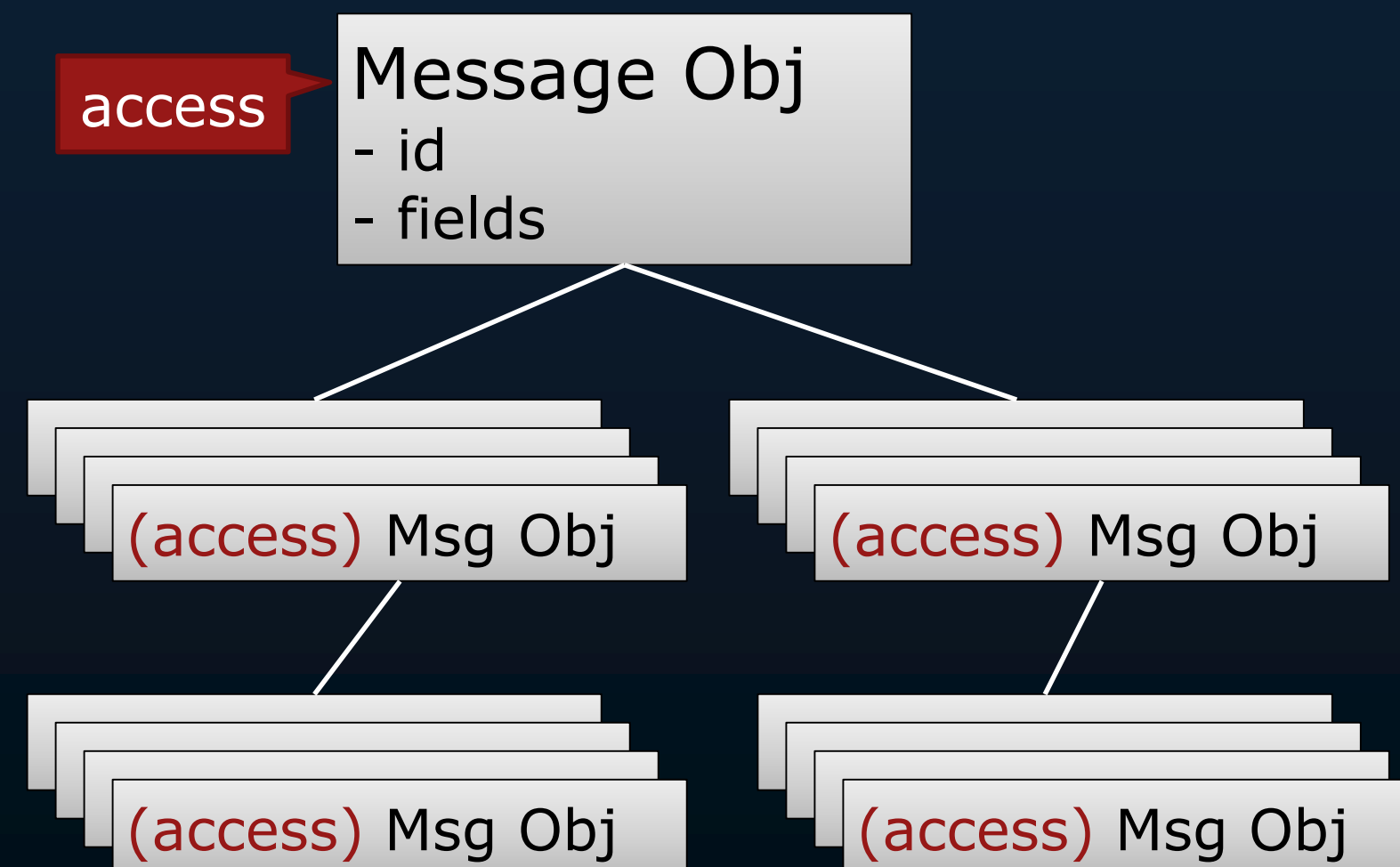
- Complex data model
- Heavy use of pointers
 - dynamic dispatch
 - reduce copying

Ada

- Retain data model
- Same use of pointers
 - dynamic dispatch
 - reduce copying

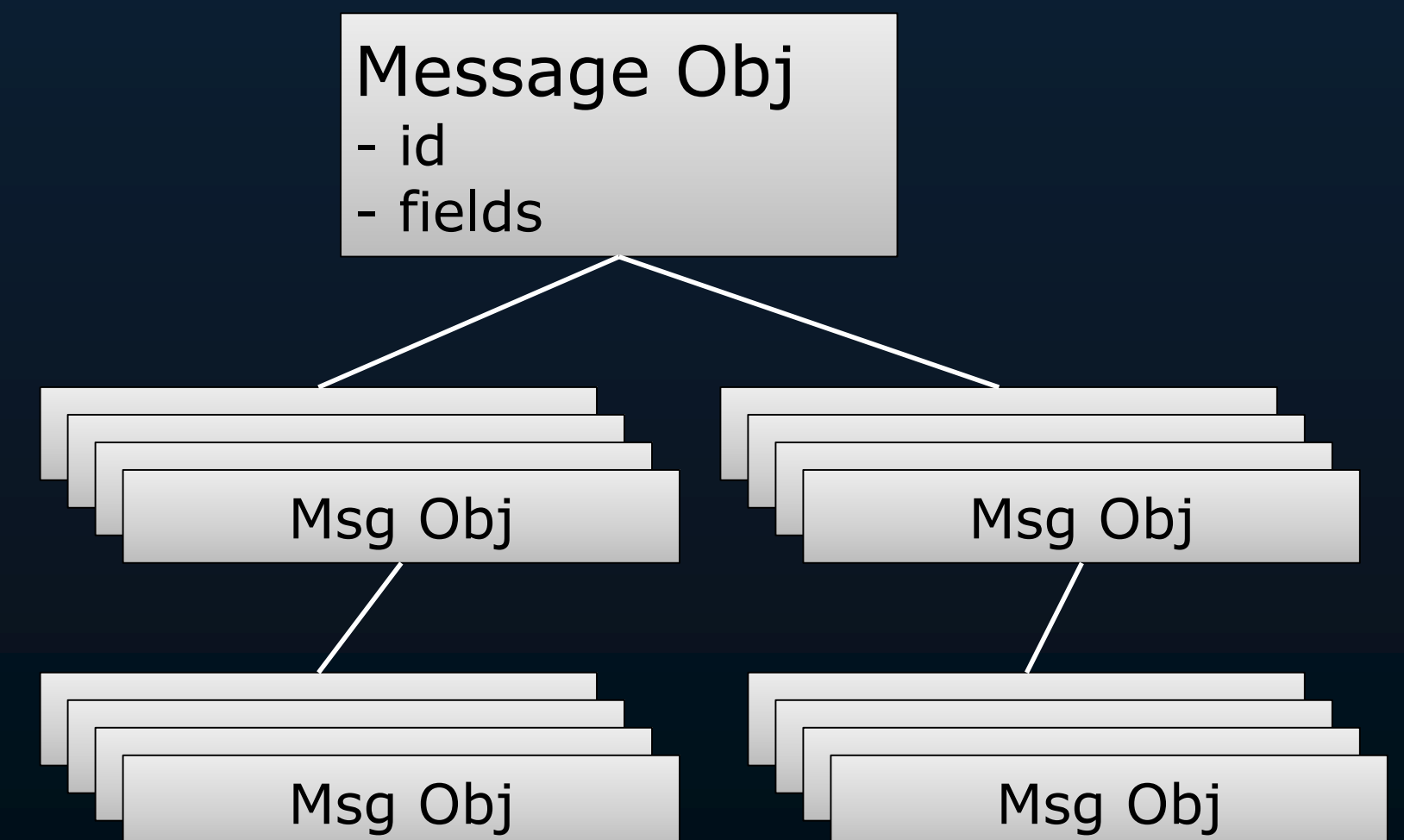
SPARK

- Restricted support for pointers
 - non-aliasing
 - not in tagged types



! →
deep copy &
replace pointers

Greedy & Expensive



Ada-SPARK Interface

C++

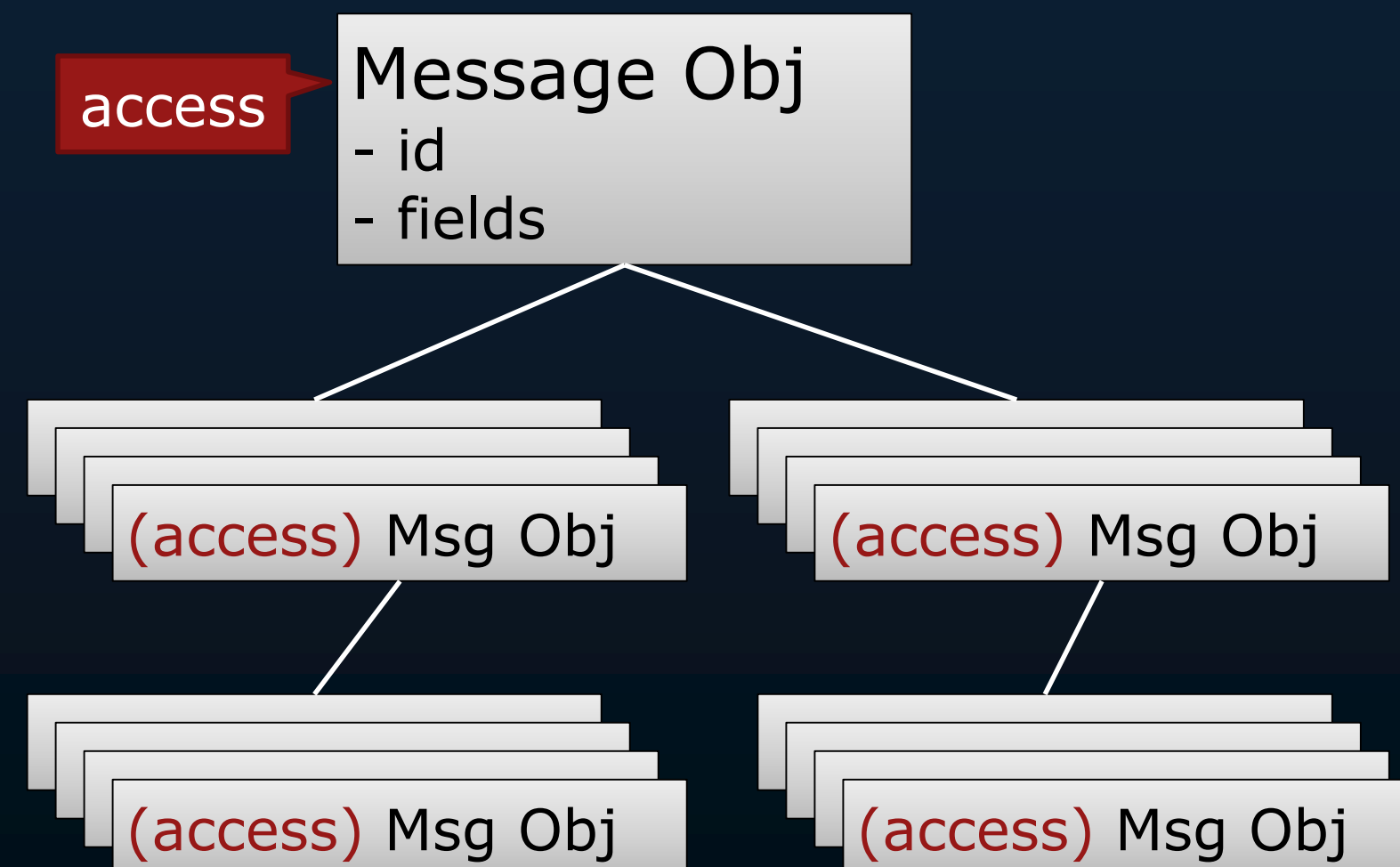
- Complex data model
- Heavy use of pointers
 - dynamic dispatch
 - reduce copying

Ada

- Retain data model
- Same use of pointers
 - dynamic dispatch
 - reduce copying

SPARK

- Restricted support for pointers
 - non-aliasing
 - not in tagged types



! →
use an adapter

Ada-SPARK Interface

C++

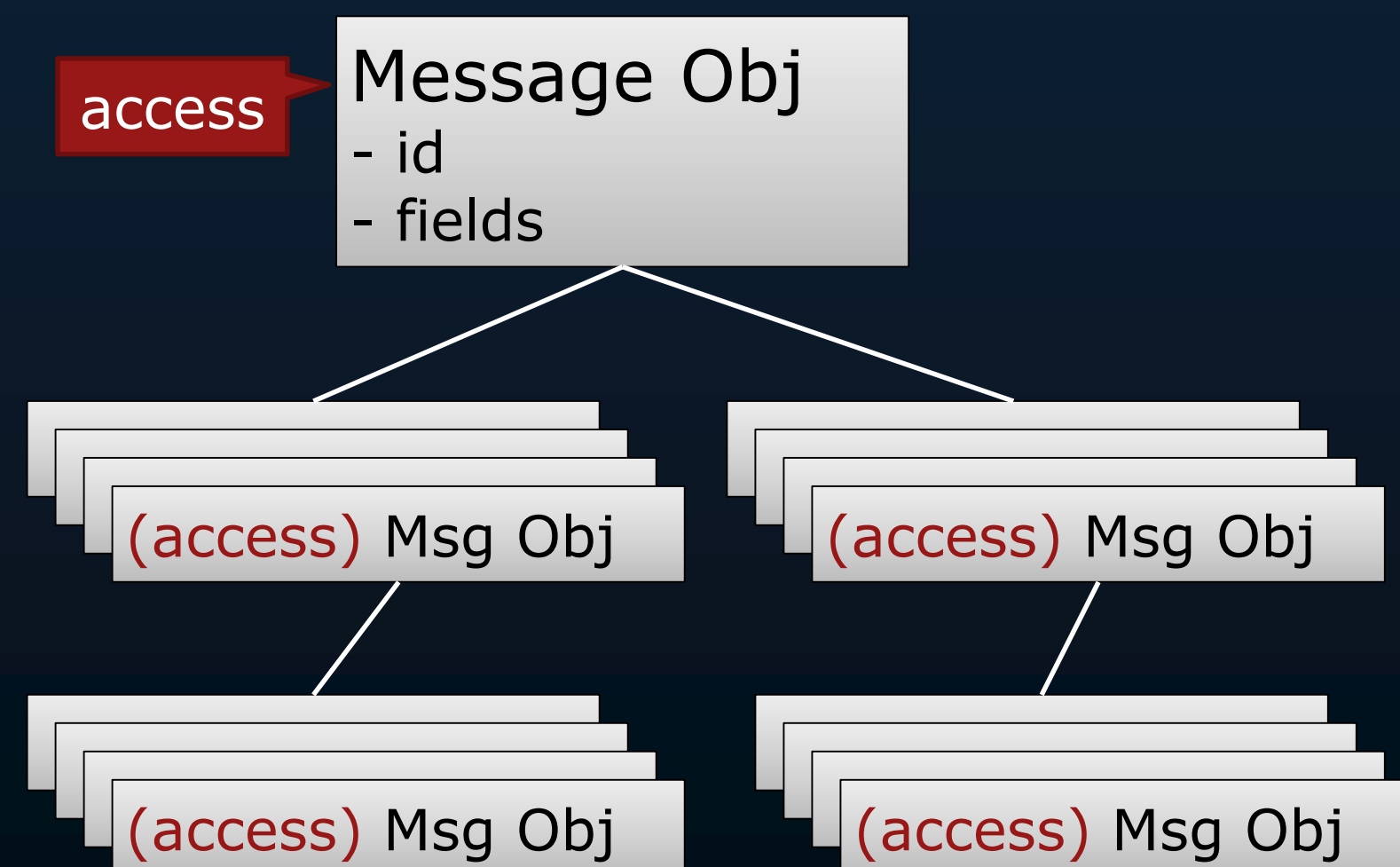
- Complex data model
- Heavy use of pointers
 - dynamic dispatch
 - reduce copying

Ada

- Retain data model
- Same use of pointers
 - dynamic dispatch
 - reduce copying

SPARK

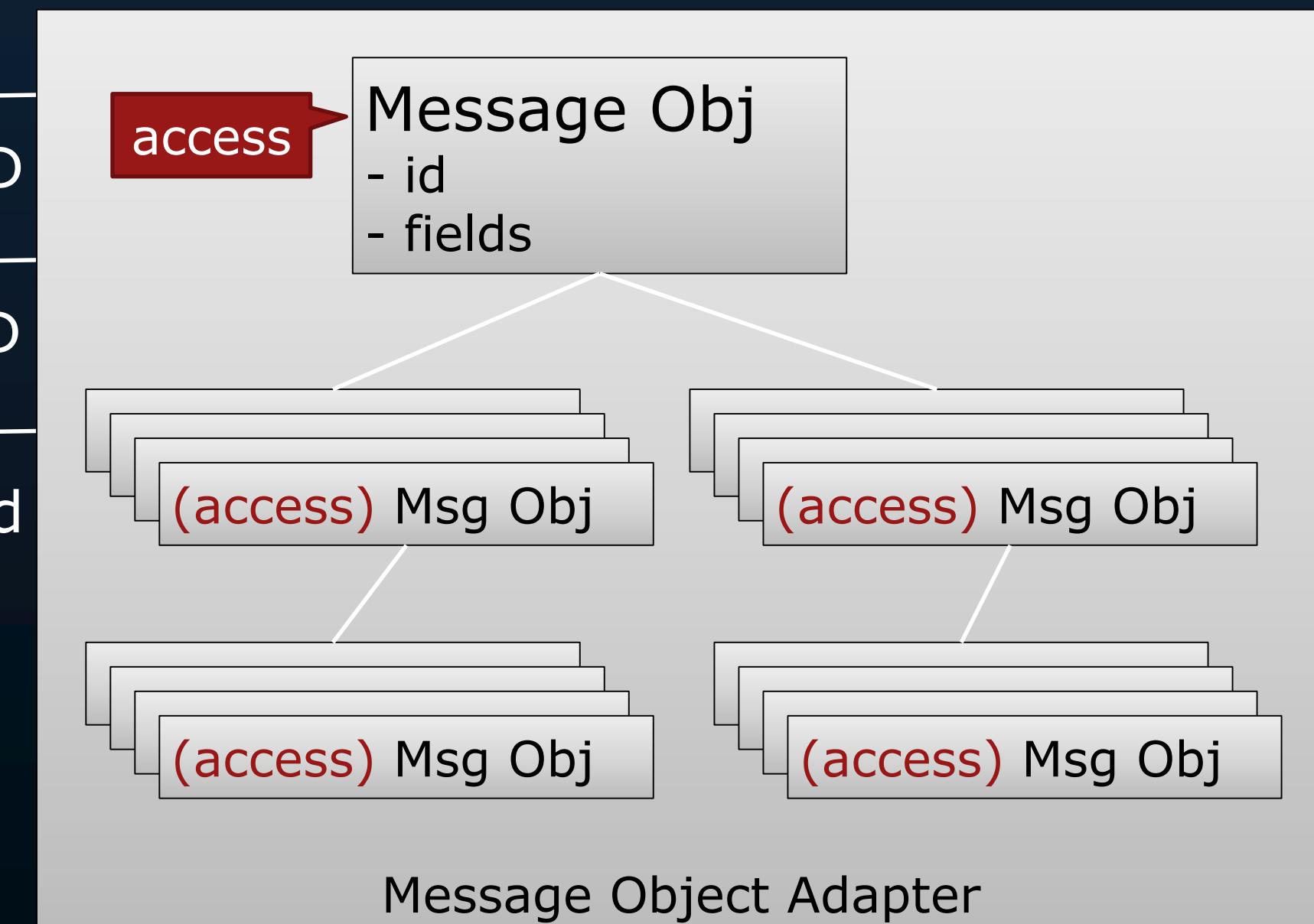
- Restricted support for pointers
 - non-aliasing
 - not in tagged types



wrap

unwrap

Get_ID
Set_ID
Get_Field



Ada-SPARK Interface

C++

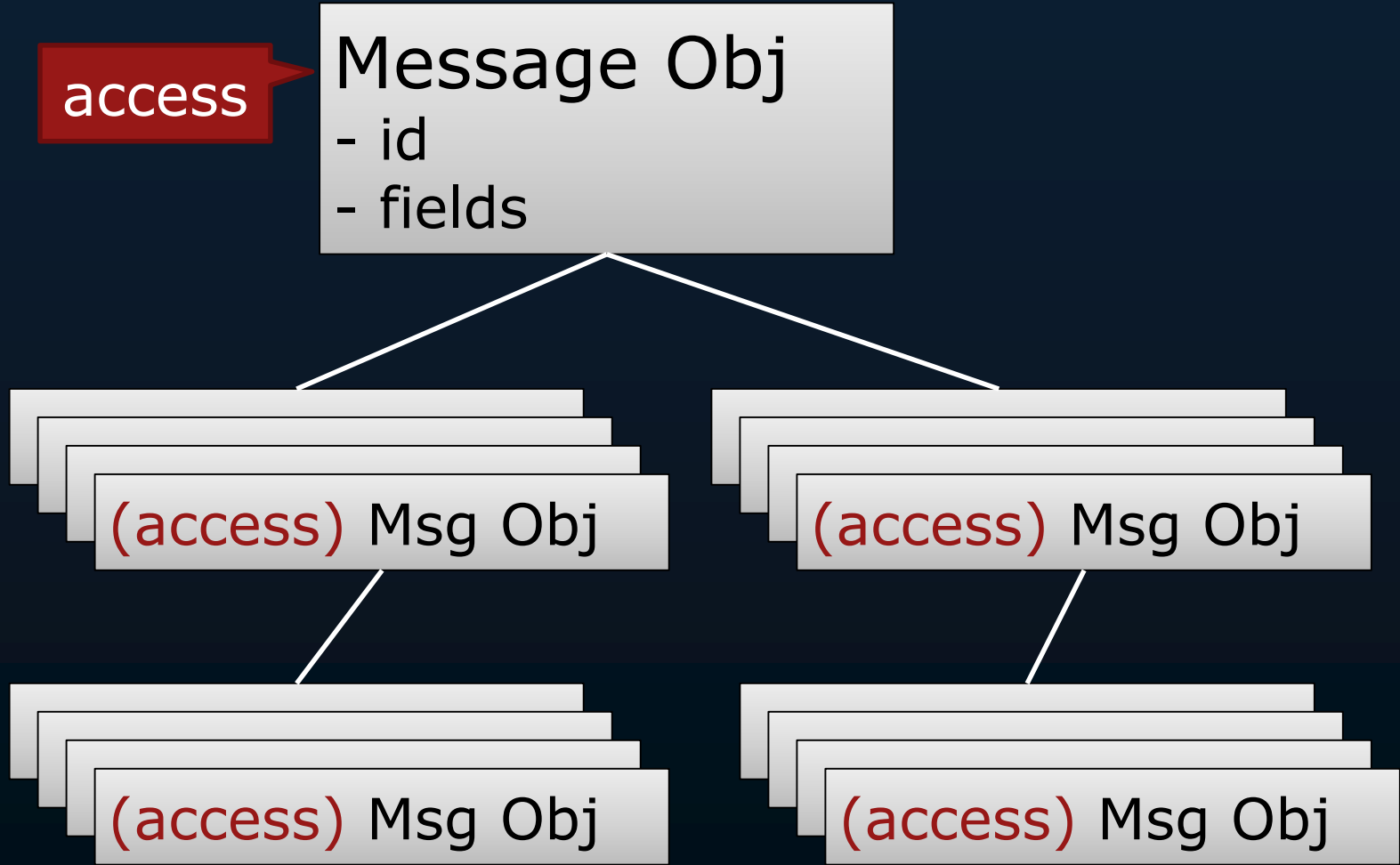
- Complex data model
- Heavy use of pointers
 - dynamic dispatch
 - reduce copying

Ada

- Retain data model
- Same use of pointers
 - dynamic dispatch
 - reduce copying

SPARK

- Restricted support for pointers
 - non-aliasing
 - not in tagged types

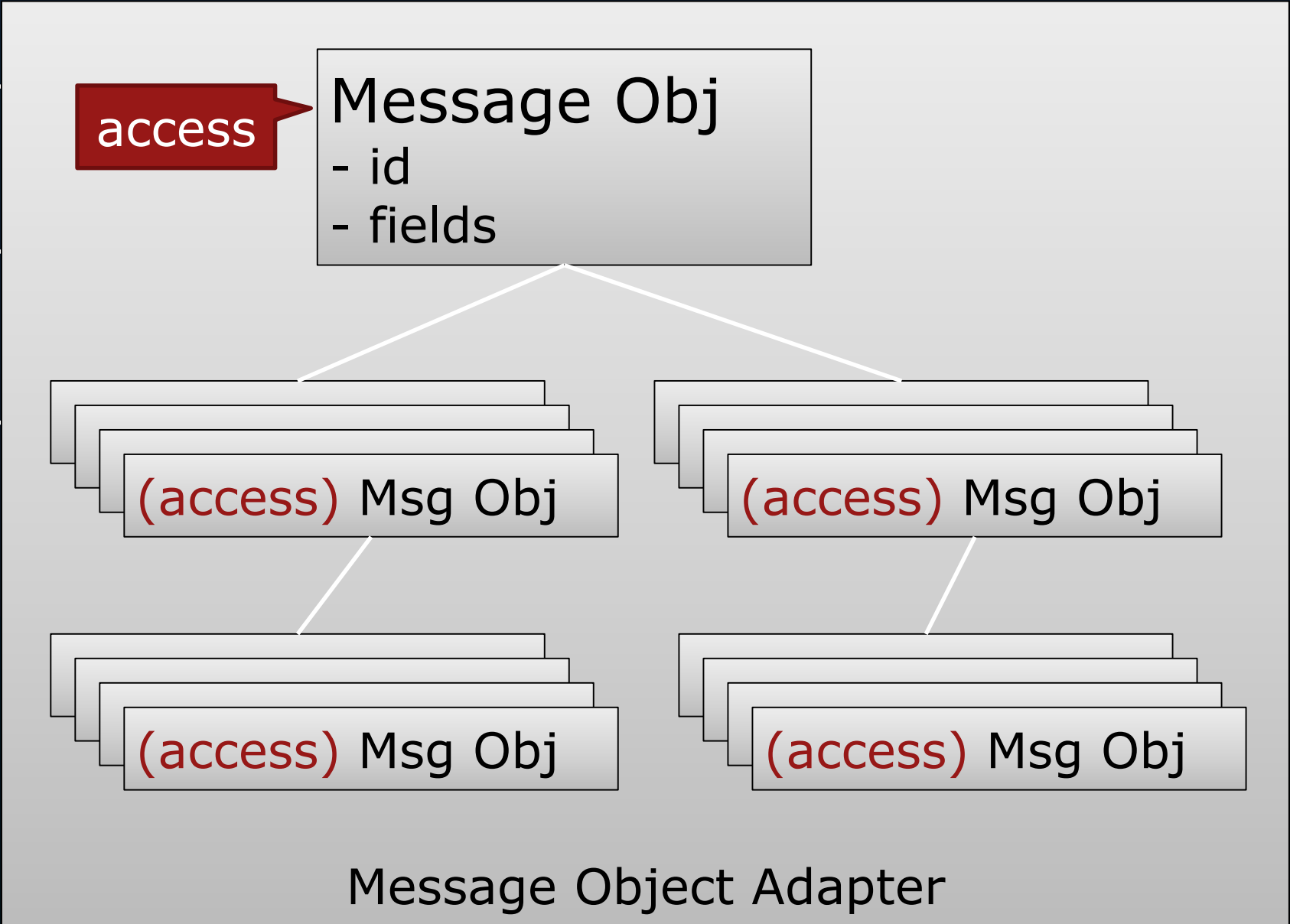


wrap

unwrap

Heavy-Weight

● Get_ID
● Set_ID
● Get_Field



Ada-SPARK Interface

C++

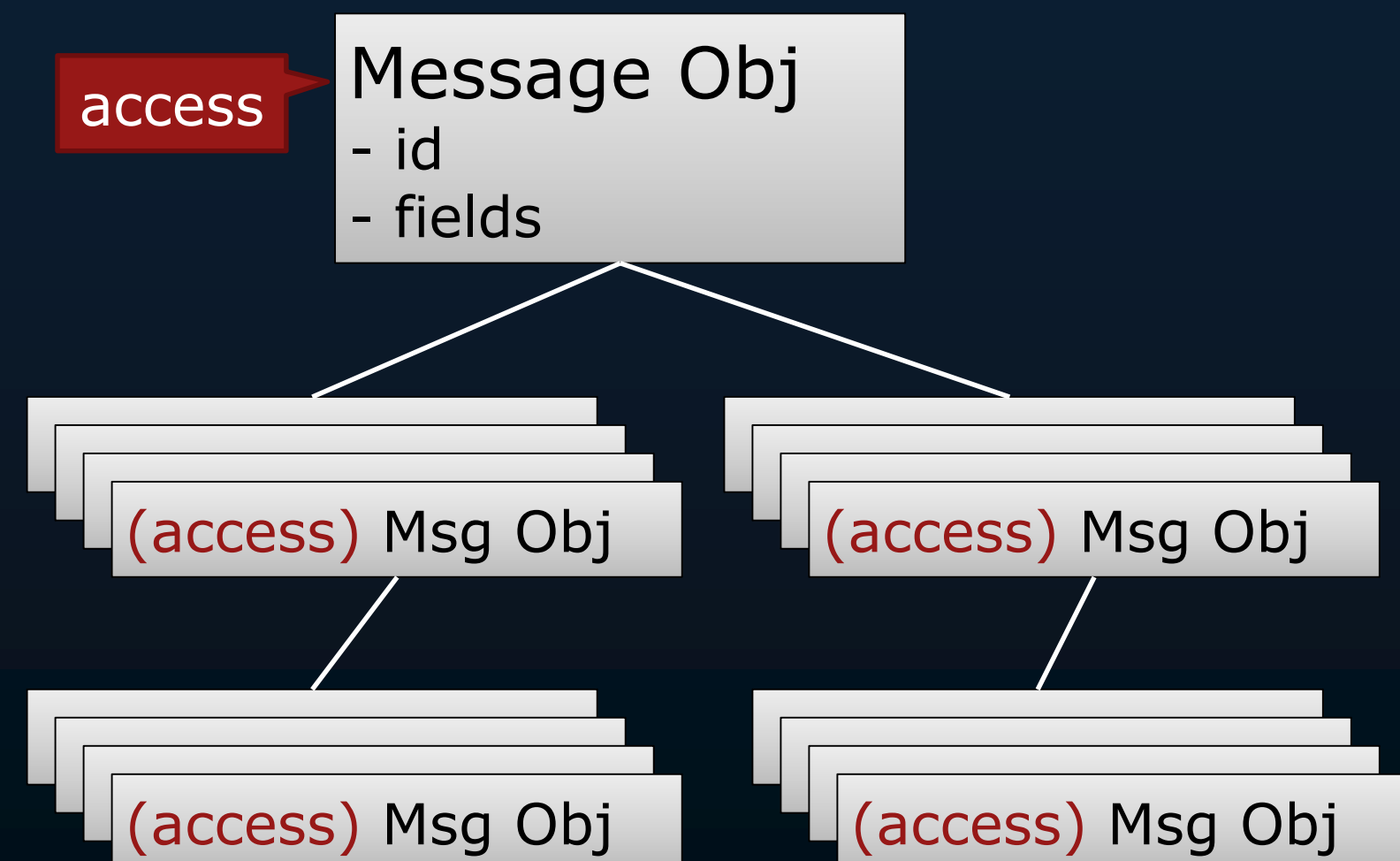
- Complex data model
- Heavy use of pointers
 - dynamic dispatch
 - reduce copying

Ada

- Retain data model
- Same use of pointers
 - dynamic dispatch
 - reduce copying

SPARK

- Restricted support for pointers
 - non-aliasing
 - not in tagged types



use a private type

Information Hiding with Private Types

```
type SPARK_Obj is private;
```

```
private
```

```
    pragma SPARK_Mode (Off);
```

```
    type SPARK_Obj is new Obj_Any;
```


Information Hiding with Private Types

```
type SPARK_Obj is private;
```

```
function Wrap (X : Obj_Any) return SPARK_Obj with SPARK_Mode  $\Rightarrow$  Off;
```

```
function Unwrap (X : SPARK_Obj) return Obj_Any with SPARK_Mode  $\Rightarrow$  Off;
```

```
private
```

```
pragma SPARK_Mode (Off);
```

```
type SPARK_Obj is new Obj_Any;
```

```
Wrap (X : Obj_Any)  $\rightsquigarrow$  (SPARK_Obj (X))
```

```
Unwrap (X : SPARK_Obj)  $\rightsquigarrow$  (Obj_Any  
(X))
```

Information Hiding with Private Types

```
type SPARK_Obj is private;  
  
function Wrap (X : Obj_Any) return SPARK_Obj with SPARK_Mode ⇒ off;  
  
function Unwrap (X : SPARK_Obj) return Obj_Any with SPARK_Mode ⇒ off;  
  
function Deref (X : SPARK_Obj) return Obj'Class;  
  
private  
    pragma SPARK_Mode (Off);  
  
type SPARK_Obj is new Obj_Any;
```

```
Deref (X : SPARK_Obj) → (X.all)
```

Ada-SPARK Interface

C++

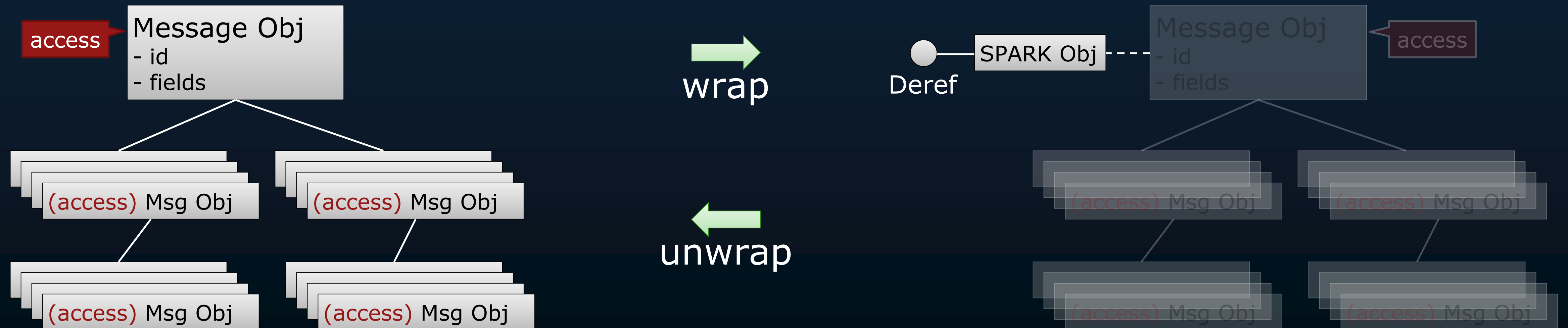
- Complex data model
- Heavy use of pointers
 - dynamic dispatch
 - reduce copying

Ada

- Retain data model
- Same use of pointers
 - dynamic dispatch
 - reduce copying

SPARK

- Restricted support for pointers
 - non-aliasing
 - not in tagged types



Ada-SPARK Interface

C++

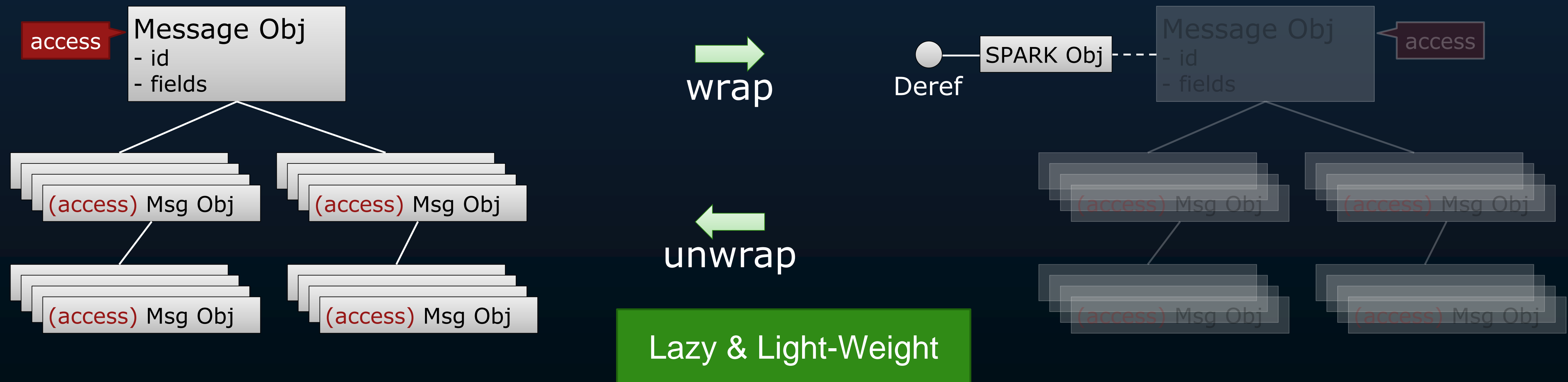
- Complex data model
- Heavy use of pointers
 - dynamic dispatch
 - reduce copying

Ada

- Retain data model
- Same use of pointers
 - dynamic dispatch
 - reduce copying

SPARK

- Restricted support for pointers
 - non-aliasing
 - not in tagged types



Proving Correctness

**Update LMCP to
automatically
generate
message types
in Ada**

**Rebuild the
service base in
Ada and bind to
ZeroMQ**

**Develop an
Ada-SPARK
interface for
sharing
message data**

**Formalize
requirements
for and prove
correctness of
the service**

Proving Correctness

- operating regions are valid
- keep-in zones are valid
- keep-out zones are valid

validator.ads

validator.adb

proved correct

- check validity of
 - operating regions
 - keep-in zones
 - keep-out zones
- provide meaningful error messages if invalid

Proving Correctness

validator.ads

[illegible]

validator.adb

[illegible]

proved correct

- ✓ **Functional Correctness**
- ✓ **Absence of Run-Time Exceptions**

Proving Correctness

validator.ads

[illegible]

validator.adb

[illegible]

proved correct

✓ Functional Correctness

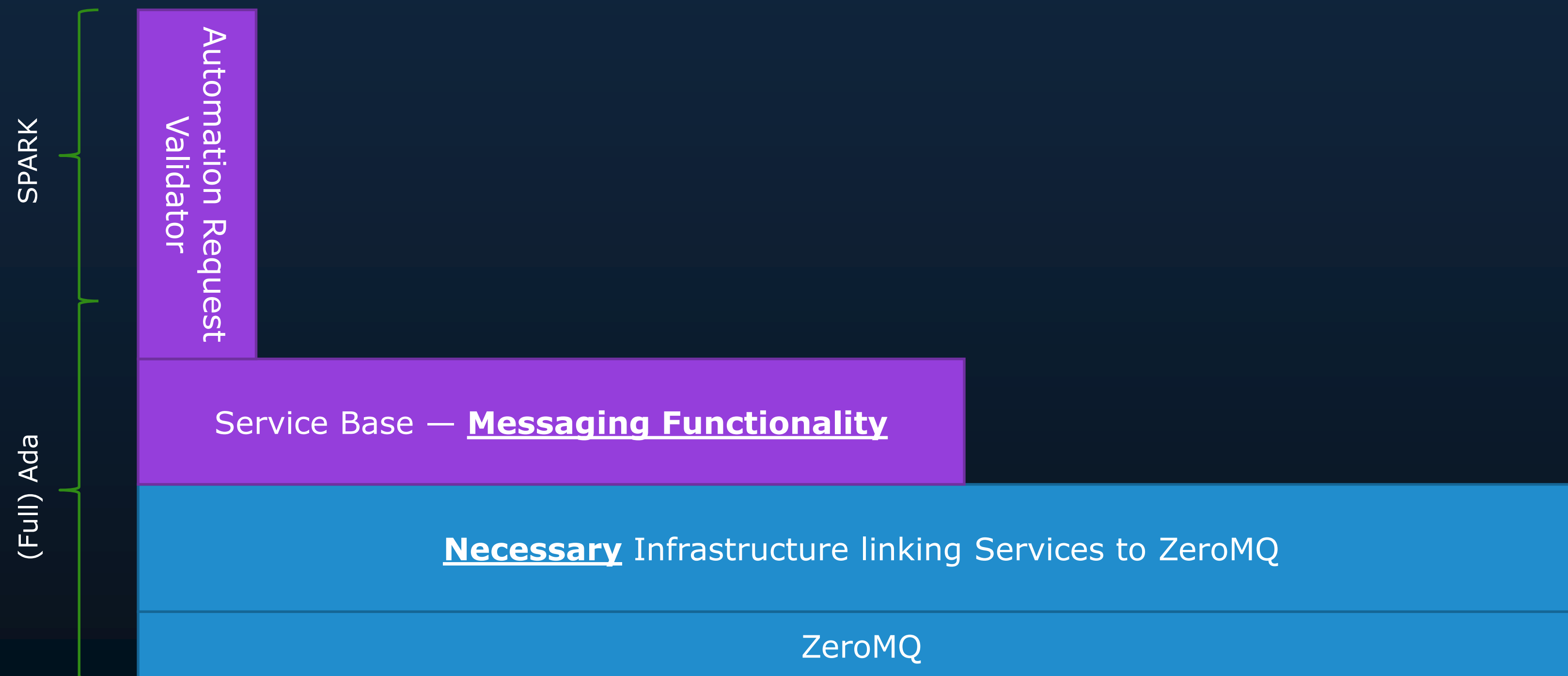
✓ Absence of Run-Time Exceptions

And We Found a Bug in the Original C++!

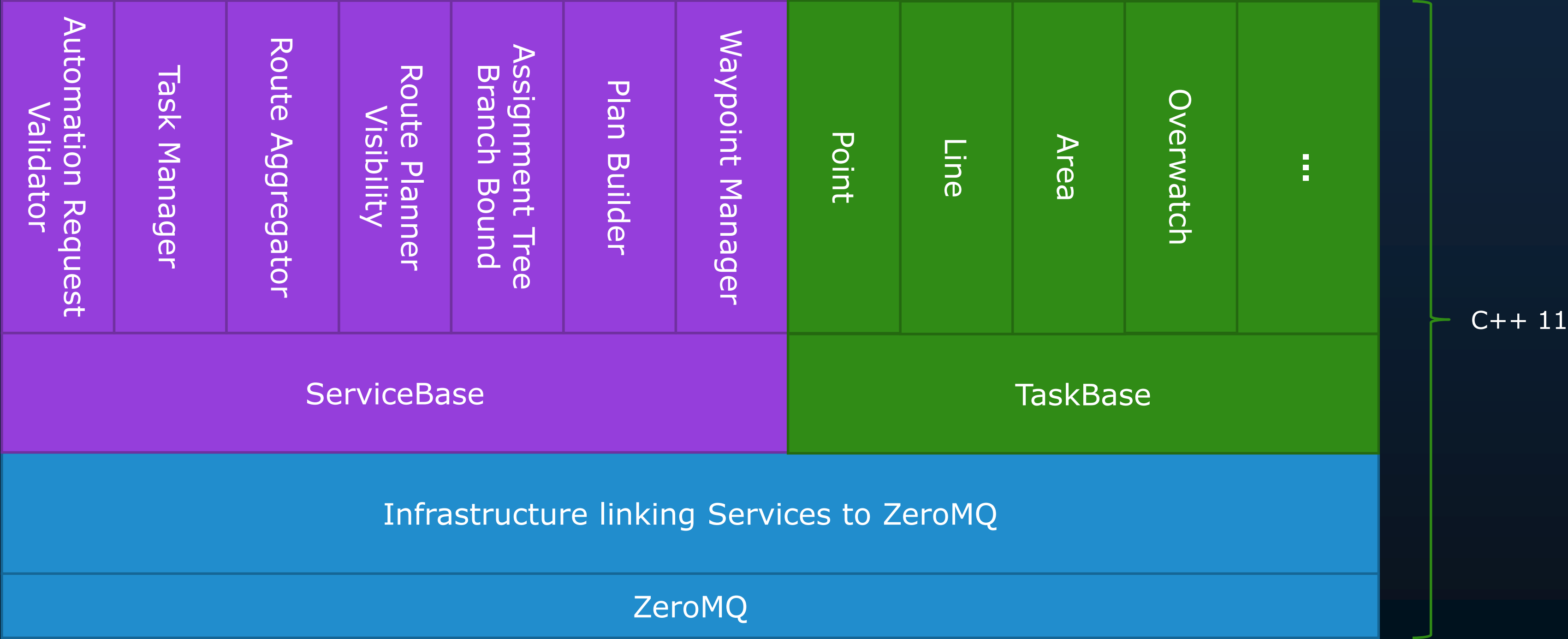
QUESTION **ANSWER**
 1. **QUESTION** **ANSWER**
 2. **QUESTION** **ANSWER**
 3. **QUESTION** **ANSWER**
 4. **QUESTION** **ANSWER**
 5. **QUESTION** **ANSWER**
 6. **QUESTION** **ANSWER**
 7. **QUESTION** **ANSWER**
 8. **QUESTION** **ANSWER**
 9. **QUESTION** **ANSWER**
 10. **QUESTION** **ANSWER**
 11. **QUESTION** **ANSWER**
 12. **QUESTION** **ANSWER**
 13. **QUESTION** **ANSWER**
 14. **QUESTION** **ANSWER**
 15. **QUESTION** **ANSWER**
 16. **QUESTION** **ANSWER**
 17. **QUESTION** **ANSWER**
 18. **QUESTION** **ANSWER**
 19. **QUESTION** **ANSWER**
 20. **QUESTION** **ANSWER**
 21. **QUESTION** **ANSWER**
 22. **QUESTION** **ANSWER**
 23. **QUESTION** **ANSWER**
 24. **QUESTION** **ANSWER**
 25. **QUESTION** **ANSWER**
 26. **QUESTION** **ANSWER**
 27. **QUESTION** **ANSWER**
 28. **QUESTION** **ANSWER**
 29. **QUESTION** **ANSWER**
 30. **QUESTION** **ANSWER**
 31. **QUESTION** **ANSWER**
 32. **QUESTION** **ANSWER**
 33. **QUESTION** **ANSWER**
 34. **QUESTION** **ANSWER**
 35. **QUESTION** **ANSWER**
 36. **QUESTION** **ANSWER**
 37. **QUESTION** **ANSWER**
 38. **QUESTION** **ANSWER**
 39. **QUESTION** **ANSWER**
 40. **QUESTION** **ANSWER**
 41. **QUESTION** **ANSWER**
 42. **QUESTION** **ANSWER**
 43. **QUESTION** **ANSWER**
 44. **QUESTION** **ANSWER**
 45. **QUESTION** **ANSWER**
 46. **QUESTION** **ANSWER**
 47. **QUESTION** **ANSWER**
 48. **QUESTION** **ANSWER**
 49. **QUESTION** **ANSWER**
 50. **QUESTION** **ANSWER**
 51. **QUESTION** **ANSWER**
 52. **QUESTION** **ANSWER**
 53. **QUESTION** **ANSWER**
 54. **QUESTION** **ANSWER**
 55. **QUESTION** **ANSWER**
 56. **QUESTION** **ANSWER**
 57. **QUESTION** **ANSWER**
 58. **QUESTION** **ANSWER**
 59. **QUESTION** **ANSWER**
 60. **QUESTION** **ANSWER**
 61. **QUESTION** **ANSWER**
 62. **QUESTION** **ANSWER**
 63. **QUESTION** **ANSWER**
 64. **QUESTION** **ANSWER**
 65. **QUESTION** **ANSWER**
 66. **QUESTION** **ANSWER**
 67. **QUESTION** **ANSWER**
 68. **QUESTION** **ANSWER**
 69. **QUESTION** **ANSWER**
 70. **QUESTION** **ANSWER**
 71. **QUESTION** **ANSWER**
 72. **QUESTION** **ANSWER**
 73. **QUESTION** **ANSWER**
 74. **QUESTION** **ANSWER**
 75. **QUESTION** **ANSWER**
 76. **QUESTION** **ANSWER**
 77. **QUESTION** **ANSWER**
 78. **QUESTION** **ANSWER**
 79. **QUESTION** **ANSWER**
 80. **QUESTION** **ANSWER**
 81. **QUESTION** **ANSWER**
 82. **QUESTION** **ANSWER**
 83. **QUESTION** **ANSWER**
 84. **QUESTION** **ANSWER**
 85. **QUESTION** **ANSWER**
 86. **QUESTION** **ANSWER**
 87. **QUESTION** **ANSWER**
 88. **QUESTION** **ANSWER**
 89. **QUESTION** **ANSWER**
 90. **QUESTION** **ANSWER**
 91. **QUESTION** **ANSWER**
 92. **QUESTION** **ANSWER**
 93. **QUESTION** **ANSWER**
 94. **QUESTION** **ANSWER**
 95. **QUESTION** **ANSWER**
 96. **QUESTION** **ANSWER**
 97. **QUESTION** **ANSWER**
 98. **QUESTION** **ANSWER**
 99. **QUESTION** **ANSWER**
 100. **QUESTION** **ANSWER**

[illegible]

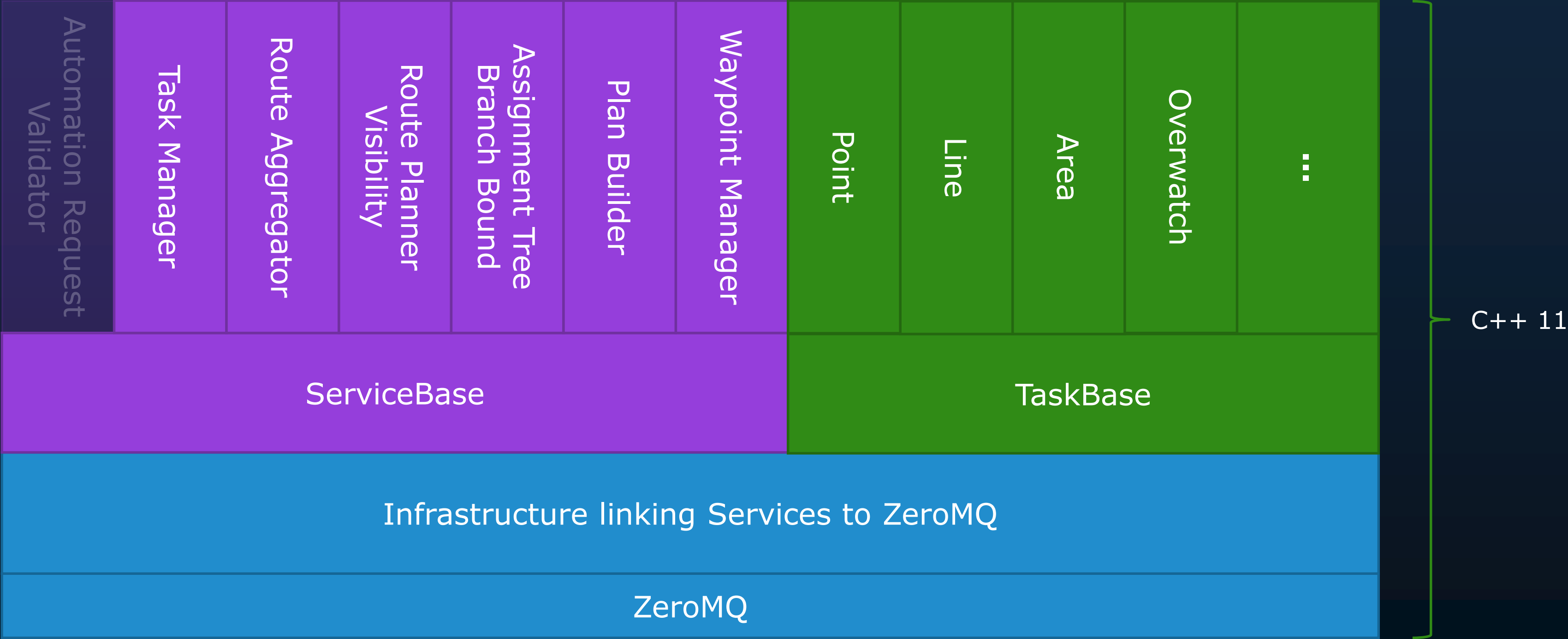
First Step



Our OpenUxAS Demo

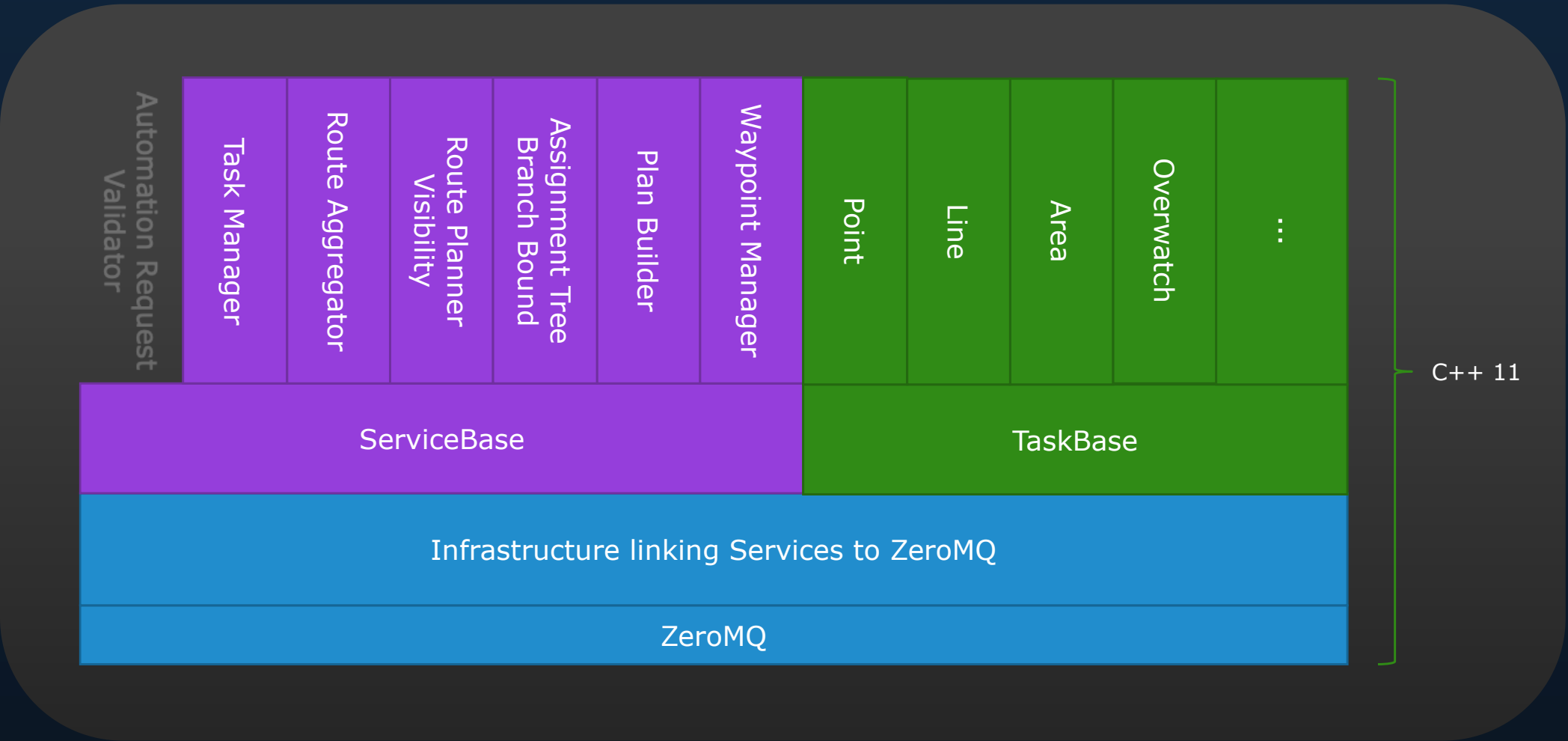


Our OpenUxAS Demo

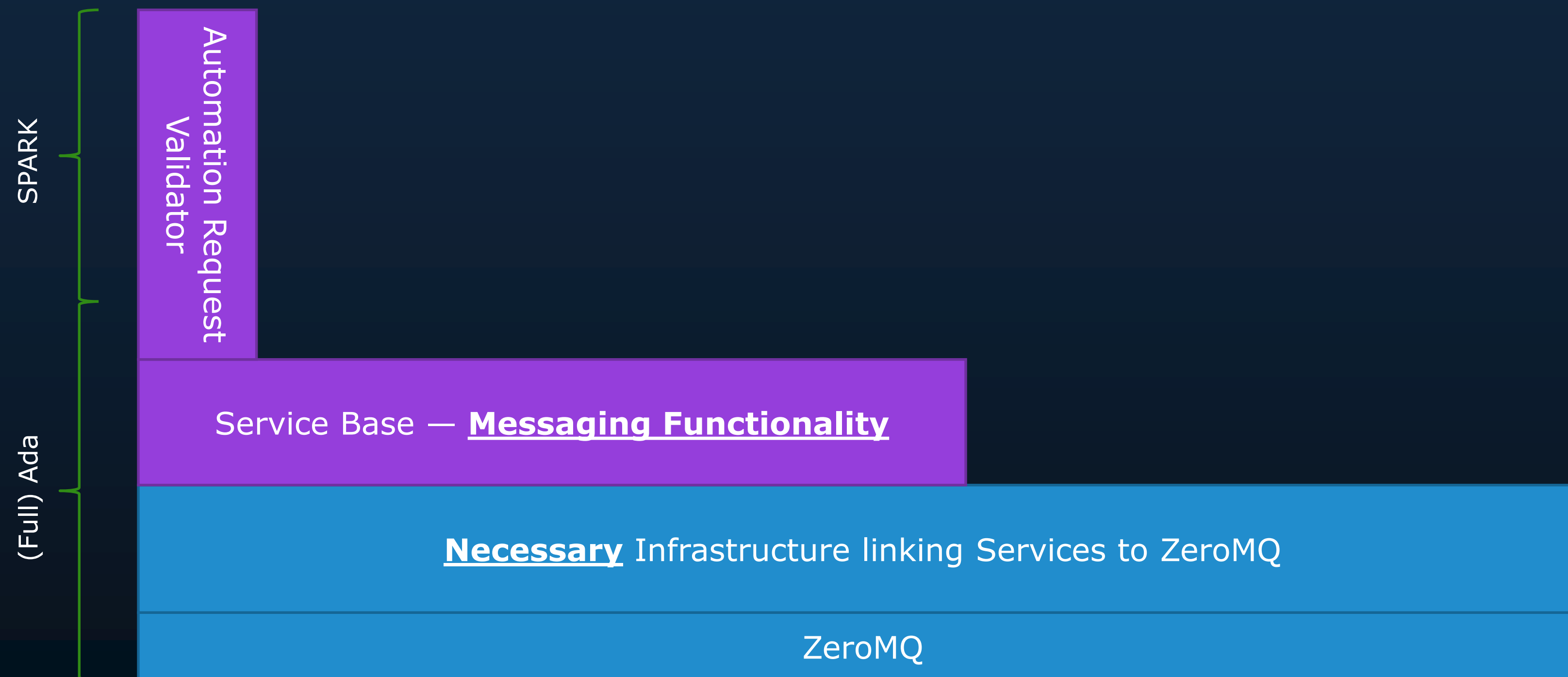


Our OpenUxAS Demo

C++ OpenUxAS Process

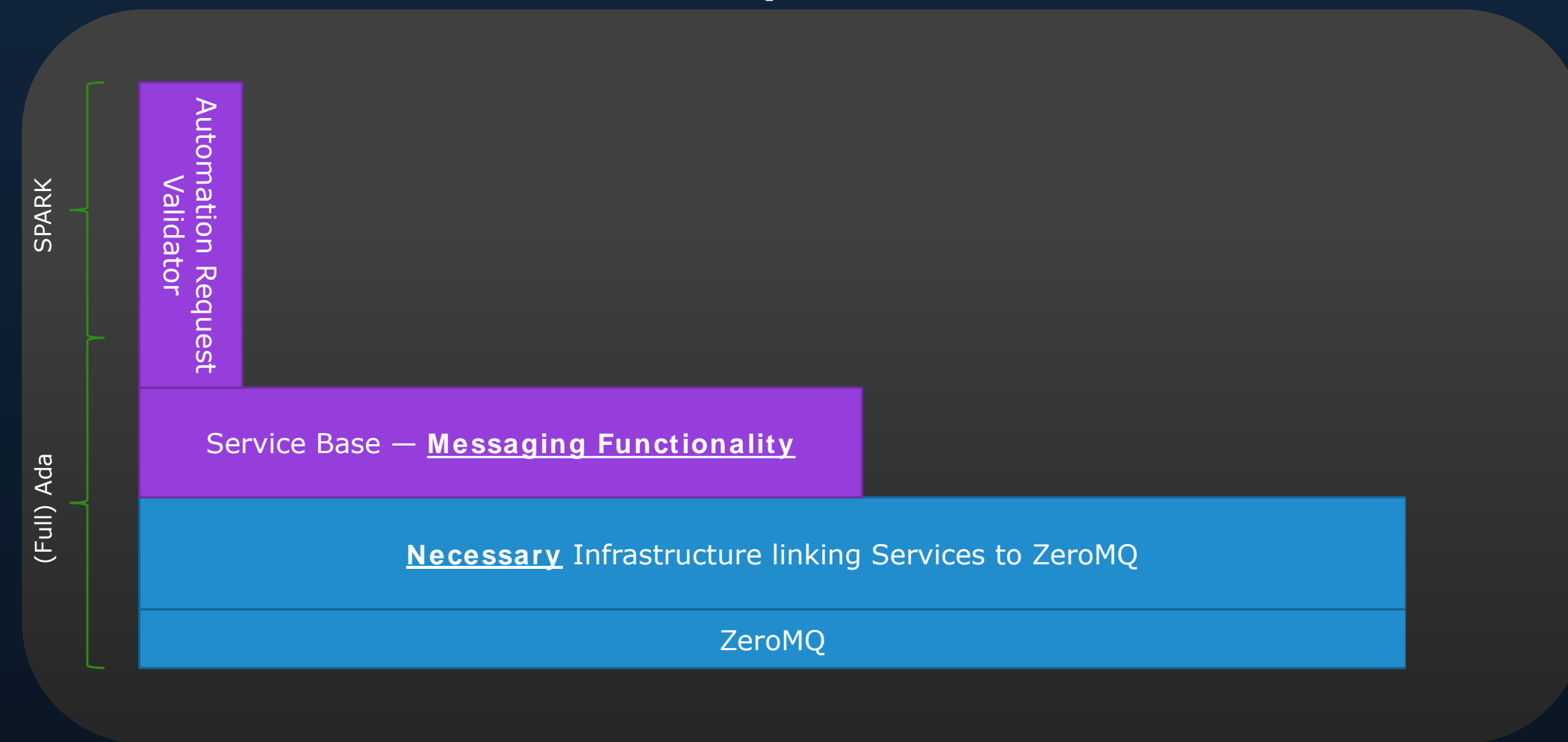


Our OpenUxAS Demo



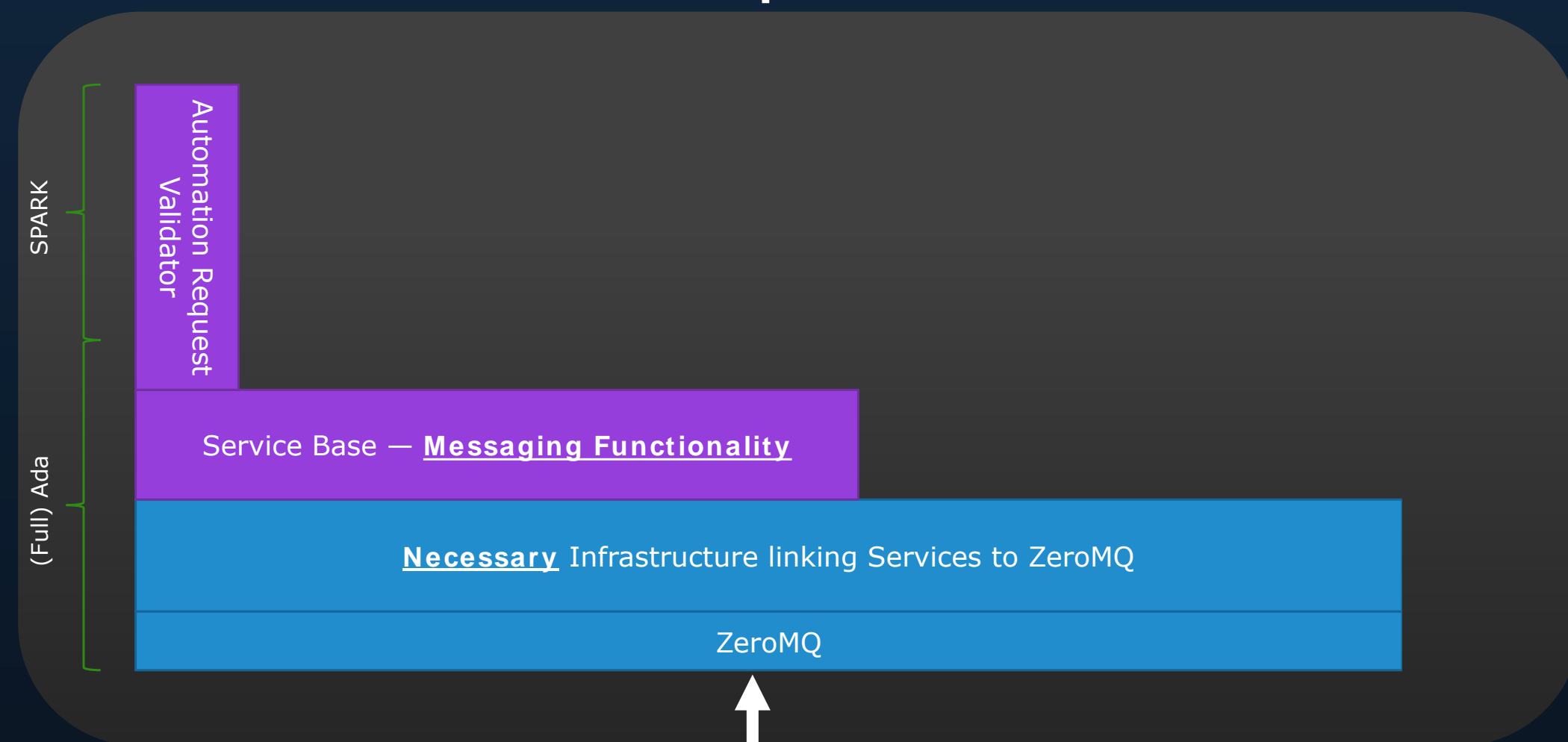
Our OpenUxAS Demo

SPARK / Ada OpenUxAS Process

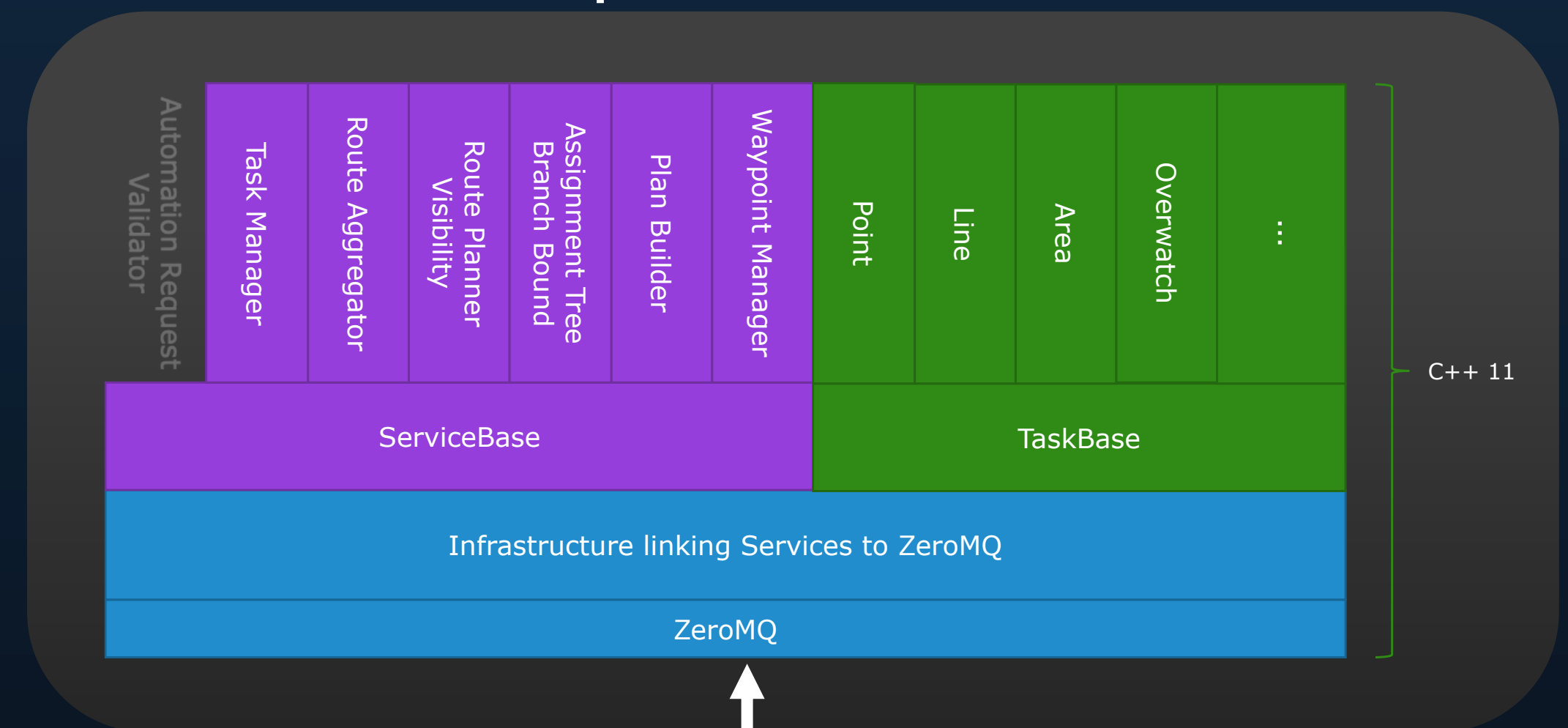


Our OpenUxAS Demo

SPARK / Ada OpenUxAS Process



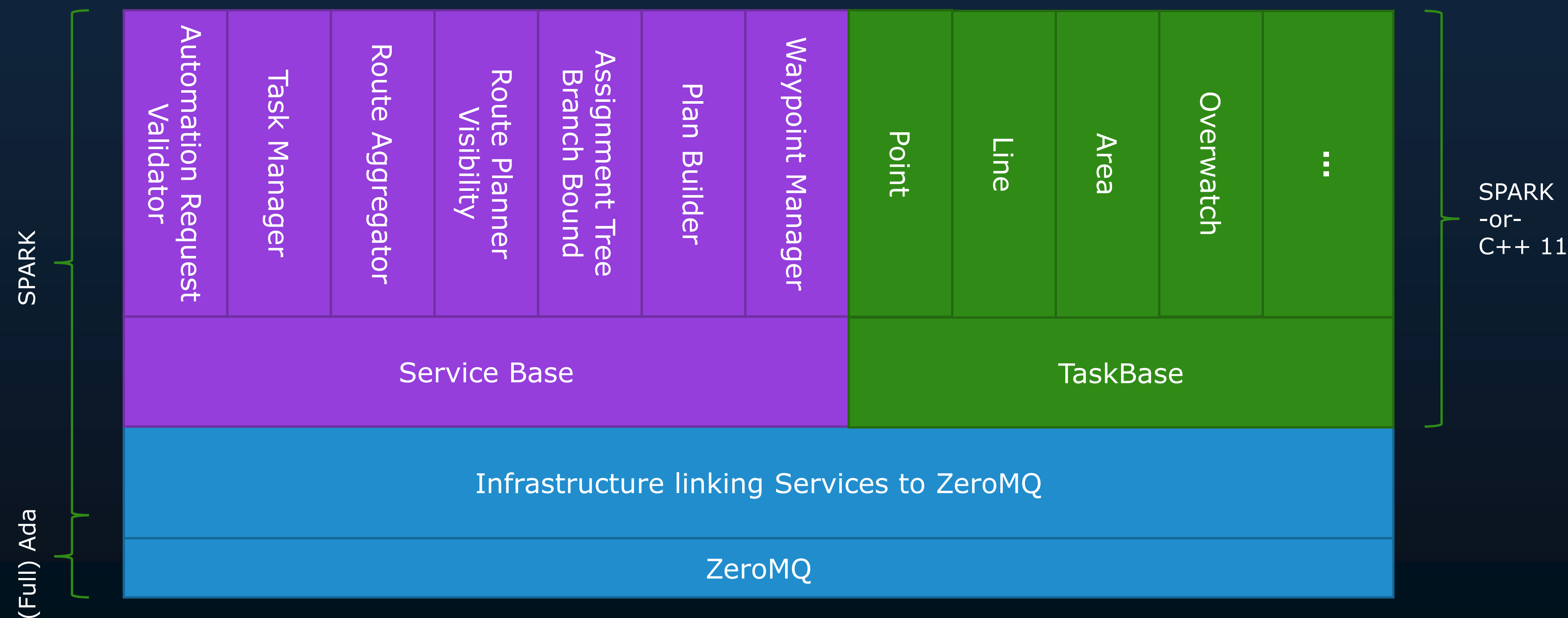
C++ OpenUxAS Process



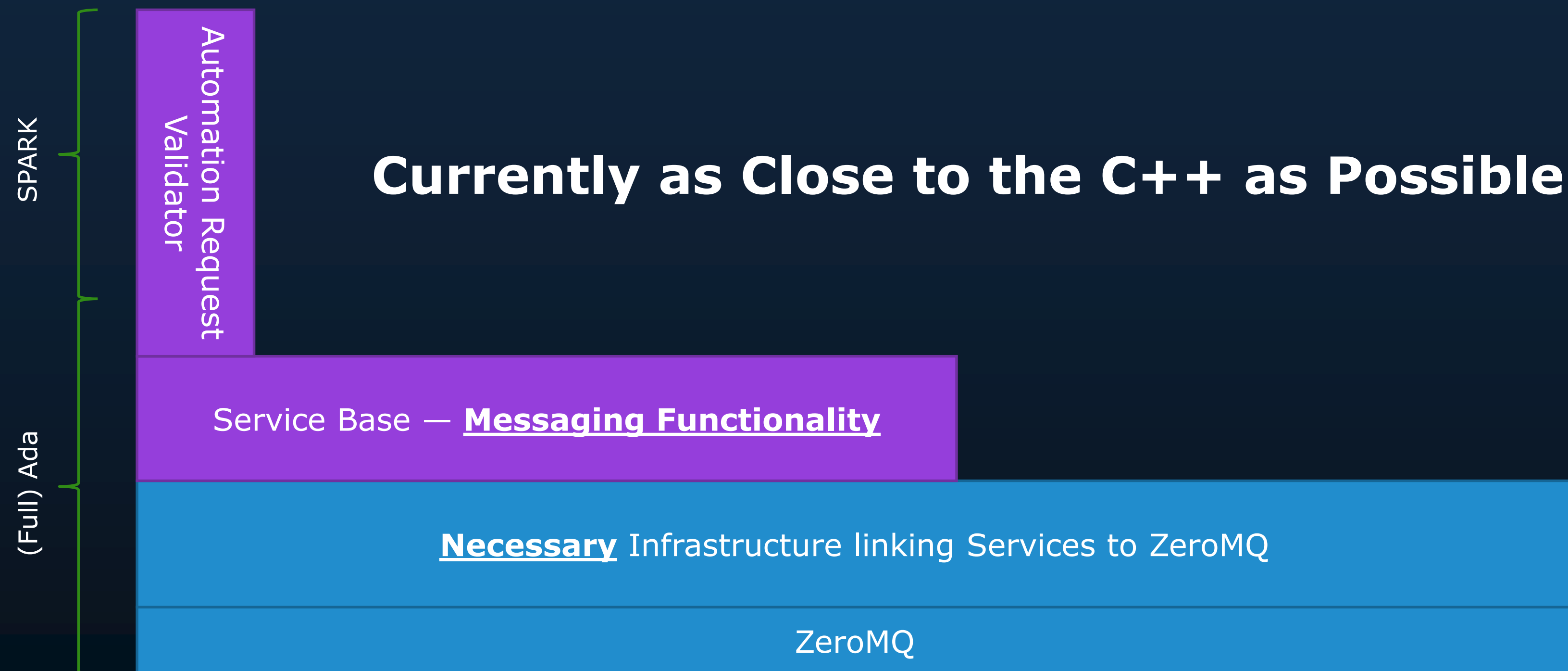
First Step

What's Next?

Goal: All Components in SPARK (and Ada)



Next Step: Refactor for Proof



Next Step: Refactor for Proof



Follow Our Progress!

github.com/AdaCore/OpenUxAS/tree/ada

AdaCore / OpenUxAS

forked from afri-rq/OpenUxAS

Watch

0

Star

0

Fork

26

Code

Pull requests 0

Projects 0

Security

Insights

Project for multi-UAV cooperative decision making

699 commits

33 branches

2 releases

11 contributors

View license

Branch: ada

New pull request

Create new file

Upload files

Find File

Clone or download

README.md



License



