



CALM

**COMMON ARCHITECTURE
LANGUAGE MODEL**

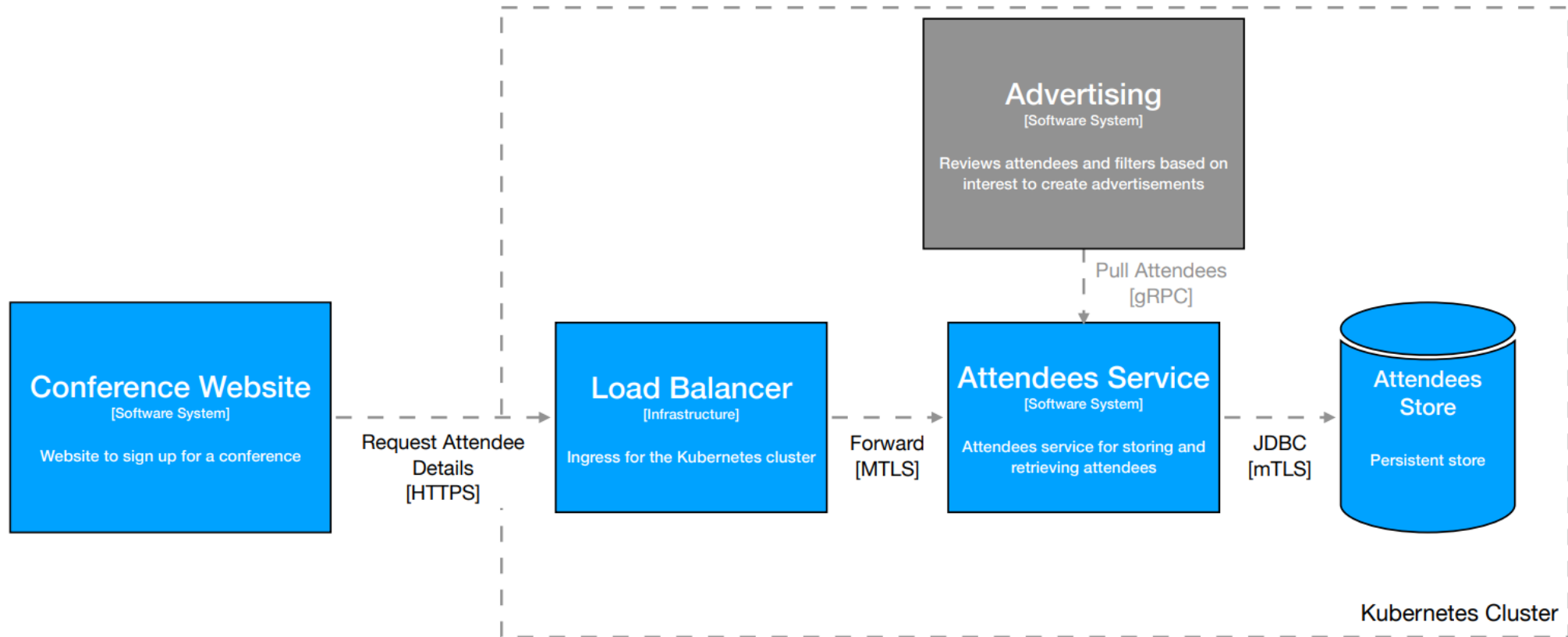
Why CALM?

- Architecture lives outside the developer workflow
- Captured on whiteboards, in Visio diagrams and PowerPoints
- Unlike requirements, code, tests, or deployments, architecture is not systematically measurable
- This leads to:
 - Uncertainty about whether what was designed is what was built.
 - Inability to detect drift over time.
 - Missed opportunities for automation and control.

What is CALM?

- A JSON Meta Schema to describe system architectures
 - Nodes >> Solid Boxes
 - Relationships >> Arrows + Dashed Boxes
 - Interfaces >> Features exposed by a node
 - Controls >> Non-functional requirements
 - Flows >> Movement through the logical architecture
 - Metadata >> Any additional information
 - Decorators >> Cross cutting concerns
 - Timelines >> Architecture changes through time

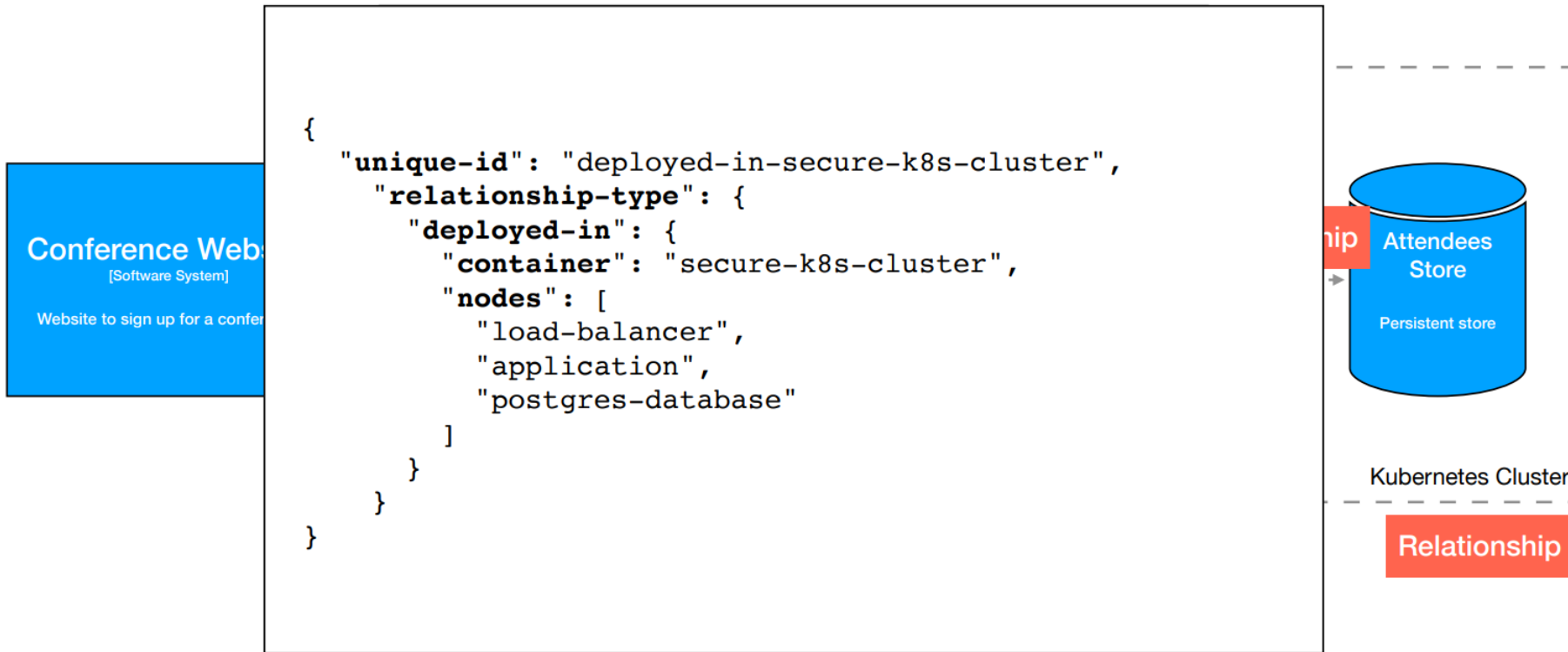
CALM – Sample Architecture



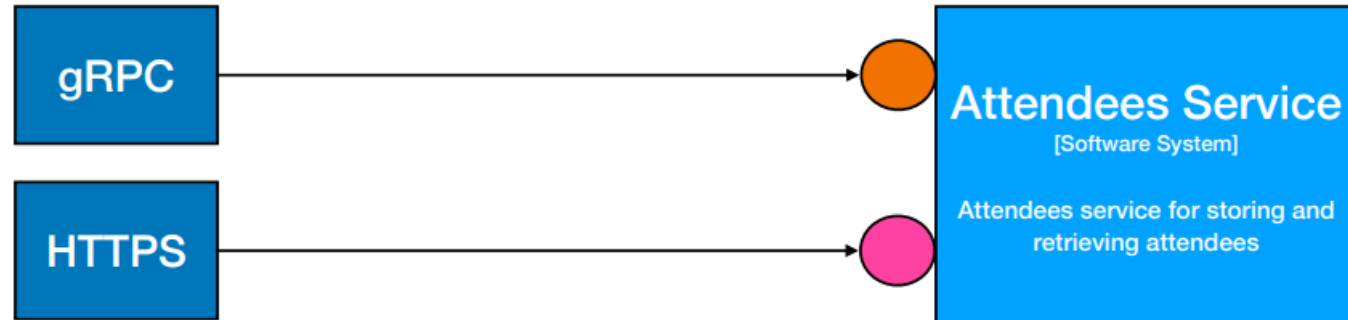
CALM - Nodes



CALM - Relationships



CALM - Interfaces



```
{  
  "unique-id": "attendees-store",  
  "name": "Attendees Store",  
  "description": "Persistent storage for attendees",  
  "node-type": "database",  
  "interfaces": [  
    {  
      "unique-id": "database-image",  
      "image": "[[ IMAGE ]]"  
    },  
    {  
      "unique-id": "database-port",  
      "port": -1  
    }  
  ]  
}, ...
```

CALM - Controls

```
{
  "$schema": "http://json-schema.org/draft/2020-12/schema",
  "title": "NIST Document Metadata",
  "type": "object",
  "properties": {
    "documentNumber": {
      "type": "string",
      "description": "The official NIST document number, e.g., 'NIST SP 800-207'"
    },
    "title": {
      "type": "string",
      "description": "The full title of the NIST document"
    },
    "status": {
      "type": "string",
      "enum": ["Final", "Draft", "Superseded", "Withdrawn"],
      "description": "The current status of the document"
    }
  },
  ...
}
```

CALM - Flows

```
"flows": [  
  {  
    "unique-id": "conference-registration-flow",  
    "name": "Conference Registration Flow",  
    "description": "Complete user journey for registering for the conference, from initial web access through data  
persistence",  
    "transitions": [  
      {  
        "relationship-unique-id": "conference-website-load-balancer",  
        "sequence-number": 1,  
        "summary": "User submits conference registration form via the website",  
        "direction": "source-to-destination"  
      },  
      {  
        "relationship-unique-id": "load-balancer-attendees",  
        "sequence-number": 2,  
        "summary": "Load balancer forwards registration request to attendees service",  
        ...  
      }  
    ]  
  }  
]
```

CALM - Decorator

```
{
  "$id": "https://calm.finos.org/release/1.2/meta/kubernetes.decorator.schema.json",
  "title": "CALM Kubernetes Deployment Decorator Schema",
  "type": "object",
  "properties": {
    "data": {
      "type": "object",
      "properties": {
        "kubernetes": {
          "type": "object",
          "properties": {
            "helm-chart": { "type": "string", "pattern": "^[a-z0-9-]+:[0-9]+\\.?[0-9]+\\.?[0-9]+$" },
            "cluster": { "type": "string" },
            "namespace": { "type": "string" }
          },
          "required": ["helm-chart", "cluster"],
          "additionalProperties": false
        }
      },
      "required": ["kubernetes"]
    }
  }
  ...
}
```

CALM - Timeline

```
{
  "current-moment": "v2-separated-db",
  "moments": [ {
    "unique-id": "v1-monolith",
    "node-type": "moment",
    "name": "Initial Monolithic Architecture",
    "description": "The original monolithic application serving all functionality",
    "valid-from": "2020-01-15",
    "details": { "detailed-architecture": "architectures/v1-monolith.json" },
    "adrs": [ https://github.com/org/repo/blob/main/docs/adr/001-initial-architecture.md ]
  }, {
    "unique-id": "v2-separated-db",
    "node-type": "moment",
    "name": "Database Separation",
    "description": "Separated the database into dedicated instances for improved scalability",
    "valid-from": "2022-06-20",
    "details": { "detailed-architecture": "architectures/v2-separated-db.json" },
    "adrs": [ https://github.com/org/repo/blob/main/docs/adr/005-database-separation.md ]
  } ],
  "metadata": {
    "title": "Payment System Architecture Timeline",
    "description": "Evolution of the payment processing system from monolith to event-driven microservices",
    "owner": "Platform Architecture Team",
  }
}
```

CALM – Patterns + Tools

- Patterns
 - Reusable, pre-approved, opinionated architectures
 - Drive standardisation + automation
 - The **superpower** of CALM
- A set of tools for working with CALM
 - CLI (published to NPM) >> generate architectures from patterns, validate architectures, create human and machine readable documents
 - CALM Hub >> NPM for CALM Artefacts, Visualisation
 - CALM Copilot Chat Mode >> Make Copilot your Architecture SME

The image shows two screenshots related to CALM. The top screenshot is a terminal window displaying the help text for the CALM CLI tool. The bottom screenshot is a screenshot of the CALM Visualizer web application, showing an architecture diagram and node details.

```
A set of tools for interacting with the Common Architecture Language Model (CALM)

Options:
-V, --version      output the version number
-h, --help         display help for command

Commands:
generate [options] Generate an architecture from a CALM pattern file.
validate [options] Validate that an architecture conforms to a given CALM
                    pattern.
server [options]   Start a HTTP server to proxy CLI commands. (experimental)
template [options] Generate files from a CALM model using a Handlebars
                    template bundle
docify [options]   Generate a documentation website off your CALM model
help [command]     display help for command
```

The bottom screenshot shows the CALM Visualizer web application. The interface includes a "Hub" and "Visualizer" section, a "Relationship Descriptions" and "Node Descriptions" toggle, and an "Upload Architecture" button. The architecture diagram shows a "Conference Website (webserver)" node connected to a "Load Balancer (network)" node, which is part of a "Kubernetes Cluster". The "Load Balancer" node is connected to an "Attendee Service (network)" node, which is also part of the "Kubernetes Cluster". The "Attendee Service" node is connected to an "Attendee Store (database)" node, which is also part of the "Kubernetes Cluster". The "Attendee Store" node is connected to an "Attendee Store (database)" node, which is also part of the "Kubernetes Cluster".

Node Details

```
{
  id: "load-balancer",
  name: "Load Balancer",
  description: "The attendees service, or a placeholder for another application",
  type: "network",
  interfaces: [
    {
      unique-id: "load-balancer-host-port",
      host: "[[ HOST ]]",
      port: -1
    }
  ],
  parent: "k8s-cluster"
}
```

Where are we?

- CALM JSON Meta Schema & CLI published v1.0 in August 2025
 - v1.1 introduced in November 2025 and v1.2 in February 2026
 - Install via NPM (`npm install -g @finos/calm-cli`) or Homebrew (`brew install calm-cli`)
- VS Code Extension published in October 2025
 - <https://marketplace.visualstudio.com/items?itemName=FINOS.calm-vscode-plugin>

AI Support

- Turn your CALM Model into a peer architect with CALM AI Support

| LLM Supported | IDE |
|---------------------------------|-------------------|
| LLM available in Github Copilot | VSCode |
| Claude family of LLM | KIRO |
| Claude family of LLM | Claude Code - CLI |

- Allows you to create all CALM components without typing any JSON
- Turns your model into an interactive query tool
- See <https://calm.finos.org/working-with-calm/calm-ai-tools> for more details

AI Support (cont'd)

- Use natural language to create your nodes
 - Create nodes representing the following services
- Instruct the LLM on how they should be connected
 - Create a connects relationship between node A and B, using B's interface
 - Create a deployed-on relationship between node A and container B
- Ask the LLM questions
 - Identify single points of failure?
 - What happens if Node B goes down?

The image displays two side-by-side windows from the CALM Preview application. The left window shows a JSON file named 'trading-system.architecture.json' with the following content:

```
1 {
2   "$schema": "https://calm.finos.org/release/1.1/meta/calm.json",
3   "unique-id": "trading-system",
4   "name": "Trading System",
5   "description": "Event-driven trading system for order submission, execution, and position management",
6   "nodes": [
7     {
8       "unique-id": "trader",
9       "node-type": "actor",
10      "name": "Trader",
11      "description": "Person who interacts with the trading system to submit orders, execute trades, and manage positions",
12    },
13    {
14      "unique-id": "web-gui",
15      "node-type": "system",
16      "name": "Web GUI",
17      "description": "Browser-based graphical user interface for traders to submit orders, execute trades, and manage positions",
18    },
19    {
20      "unique-id": "account-service",
21      "node-type": "service",
22      "name": "Account Service",
23      "description": "Manages account creation, updates, and user associations; validates user information against the user directory",
24    },
25    {
26      "unique-id": "trade-service",
27      "node-type": "service",
28      "name": "Trade Service",
29      "description": "Handles trade ticket submission; validates trades against security master and account information",
30    },
31    {
32      "unique-id": "position-service",
33      "node-type": "service",
34      "name": "Position Service",
35      "description": "Provides initial bootstrap data for trades and positions when user opens a new account",
36    },
37    {
38      "unique-id": "people-service",
39      "node-type": "service",
40      "name": "People Service",
41      "description": "Searches for users by name and validates user information against the user directory",
42    },
43    {
44      "unique-id": "security-master",
45      "node-type": "service",
46      "name": "Security Master",
47    }
48  ]
49 }
```

The right window shows the 'Architecture Overview' diagram, which is a complex flowchart illustrating the relationships between the nodes defined in the JSON. The diagram includes nodes such as 'Trader', 'Web GUI', 'Account Service', 'Trade Service', 'Security Master', 'Trade Processor', 'Trade Data Store', 'Position Service', 'Trade Data Store', 'People Service', and 'User Directory'. Arrows indicate various relationships like 'Interacts with', 'Subscribes to', 'Validates', and 'Queries'.

See <https://calm.finos.org/tutorials/build-a-calm-architecture/define-initial-architecture> for details

Where are we going?

- Adding more capabilities
 - Deployment & Source Control lenses
- Further use-cases:
 - Policy as Code
 - Architecture drift
 - Partner with Moderne to further developer Prethink - <https://docs.moderne.io/user-documentation/moderne-platform/getting-started/prethink/>

Like what you see?

- Get involved, we're looking for early adopters who are keen to learn CALM and provide feedback or open source contribution
 - Good First Issues: <https://github.com/finos/architecture-as-code/issues?q=state%3Aopen%20label%3A%22good%20first%20issue%22>
- CALM tutorials: <https://calm.finos.org/tutorials/>
- Two regular catch ups:
 - **Monthly Architecture as Code Working Group Meeting:** fourth Tuesday of every month @ 11:00 AM to noon Eastern
 - **Weekly Architecture as Code Office Hours:** Thursday @ 10:30AM to 11:30AM
- Head over to <https://calendar.finos.org/> to sign up



Who Am I?

- Khalid Elsawaf – Executive Director at Morgan Stanley
- Lead Architecture for Secured Financing Technology Post Trade
- Montreal Lead for Prime Brokerage & Secured Financing Technology
- Email: khalid.elsawaf@morganstanley.com
- LinkedIn: <https://ca.linkedin.com/in/khalid-elsawaf-33560811>

