

The background is dark with abstract, glowing elements. A large, curved, metallic-looking structure with a bright light source at its top right end dominates the right side. A smaller, curved, ribbed structure with a small blue square marker is in the lower right. The text is positioned on the left side of the image.

Shipping with Context: **Knowledge Graphs as the** **Backbone of AI-First** **Software Delivery**

The Context Challenge in Enterprise AI



Siloed Data & Tools

Information trapped in disparate systems, inaccessible to AI.



Lack of Domain Knowledge

Generic LLMs don't understand company-specific jargon or unique processes.



Risk of Hallucinations

Ungrounded AI can confidently generate incorrect, misleading information.



Security & Privacy Risks

Sharing sensitive data with external models poses significant compliance threats.

AI-Assisted DevOps vs. AI-Operational DevOps

AI-Assisted DevOps

- AI helps *humans* write code, configs, runbooks
- Context injected manually via prompts
- Human remains the primary safety system

AI-Operational DevOps

- AI acts *within* the delivery system
- Pulls context automatically from a shared model
- Performs checks, simulations, and policy evaluation before acting

With Knowledge Graphs,

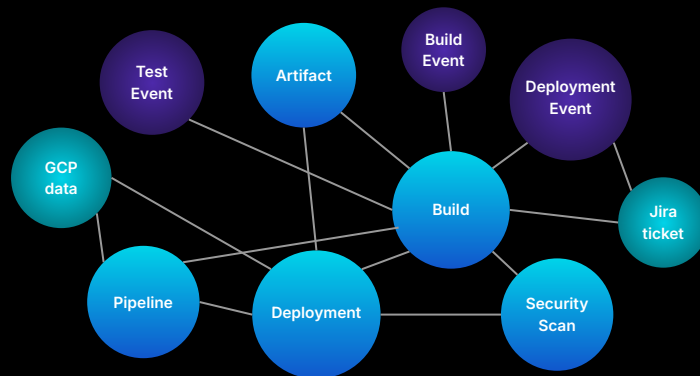
- Context becomes queryable, not conversational
- Safety moves from humans → system defaults

Anatomy of the Knowledge Graph

i.e., "Context"

Core Entities (The Nodes)

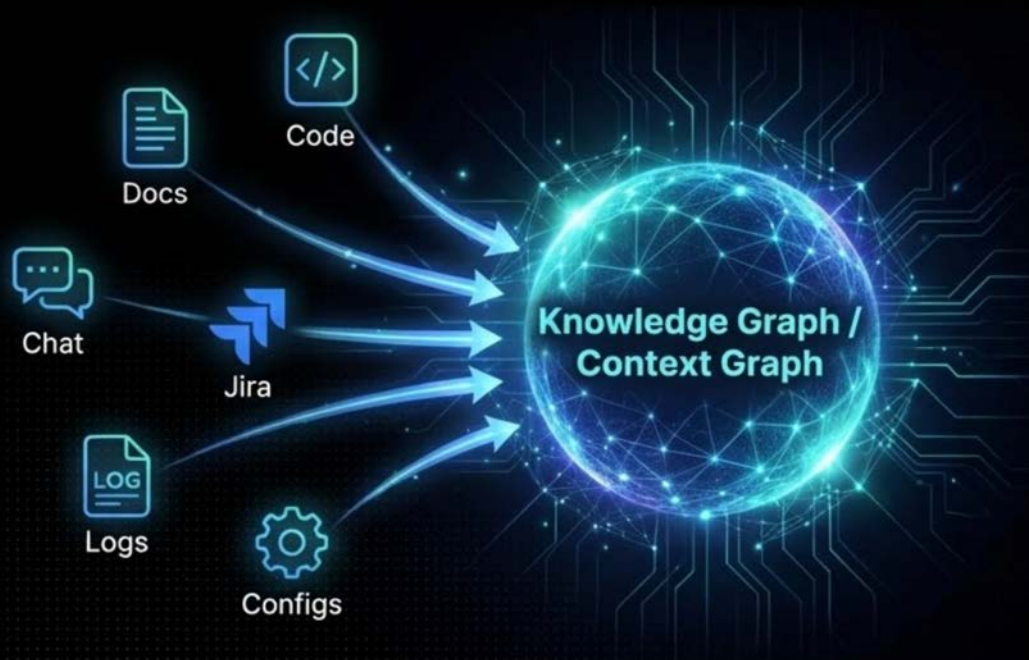
- Your data: Service, Pipeline, Build, Deployment, Incident, Policy, Cloud Cost, Owner, etc
- Your events: Build event, Deployment event, etc.
- 3rd-party data: GCP, GKE data



Critical Relationships (The Edges)

- Service -> Depends On -> Service (Blast radius)
- Deployment -> Triggered By -> Commit (Root cause)
- Service -> Owned By -> Team (Accountability)

Building the Knowledge Graph



Extract entities
(services, users, incidents)



Enrich with metadata
(owners, versions, policies)



Link relationships
(who owns what, what depends on what)



Expose via Graph API for AI agents to query

If context is the new code, knowledge graph is the compiler

Where can Knowledge Graphs fail?

Primary reason for knowledge graph to fail: **Incorrect use cases**

Incorrect / Not well-defined uses cases lead to:

Over-modeling

Beautiful ontology,
zero usage

Under-modeling

Flat data, no
meaningful
relationships

Stale data

Technically correct but
operationally wrong
graph

Demo

