EPIC GAMES · GAMES ON AWS 2026

From Creators to Cloud

Version control and content pipelines with Lore on AWS

Mattias Jansson — Technical Director, Developer Tools
Epic Games

AGENDA

Two ideas, one system underneath

01 The problem

02 Lore

03 Creators

04 Cloud

05 Your studio

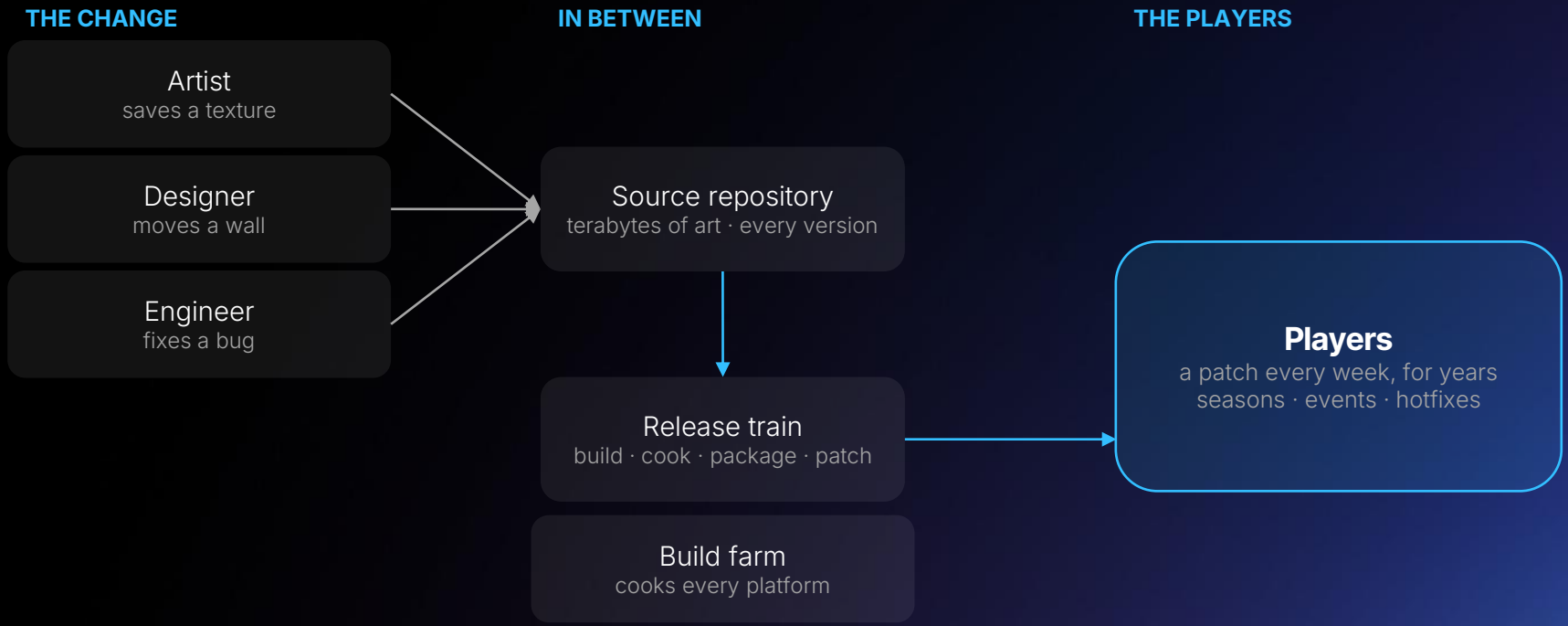
01

The problem

A live-service game ships content forever.

01 · THE PROBLEM

From a saved file to a shipped patch



01 · THE PROBLEM

Where the time and the budget go

THE TOOL

Built for text, holding binary art

THE PEOPLE

Two crews, opposite needs, one repository

THE PIPELINE

Storage and version control never share a byte

THE BILL

Every stage copies what did not change

01 · THE PROBLEM

What we needed instead

Version control built for binary art

Two workflows, one repository

One store from source to cooked output

Only real changes move



02

Lore

Open-source version control and storage, three years in production at Epic.

02 · LORE

Three ideas carry the whole system

THE ADDRESS

The address is the content

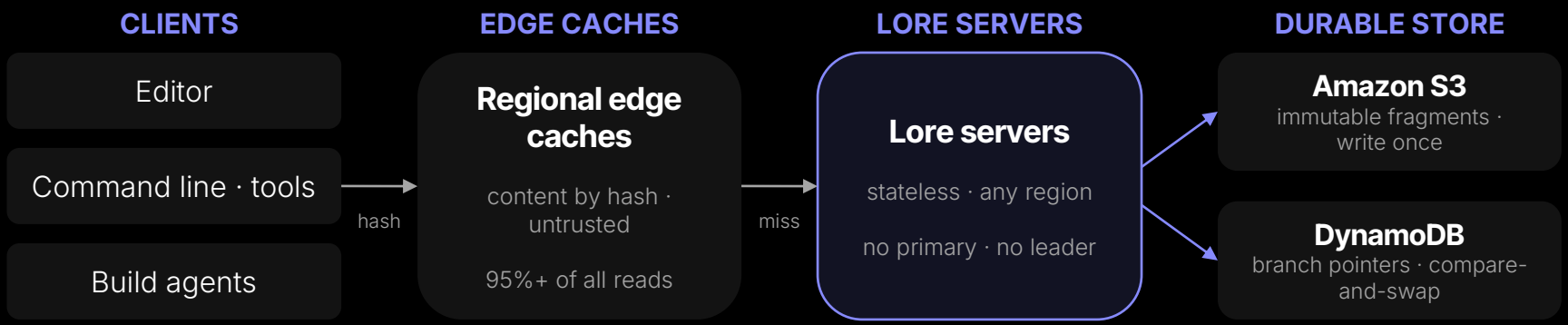
THE REVISION

A revision is the whole tree

THE BRANCH

A branch is a pointer

How it is built



Content flows through caches. The only serialised write is one pointer swap.

One store, many applications

IN-HOUSE DEVELOPMENT

Engine and Fortnite Multi-terabyte projects

CREATORS

Unreal Editor for Fortnite

BUILD PIPELINE

Epic's build and content pipeline on AWS

Source code, creator islands, cooked builds.
One address space, one set of permissions, stored once.

03

Creators

Three years of Unreal Editor for Fortnite.



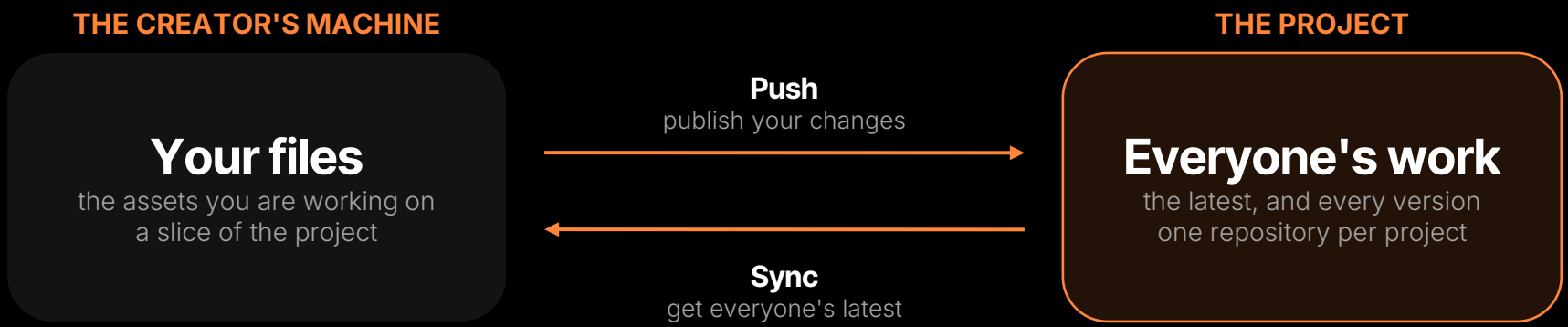
03 · CREATORS

Version control nobody had to learn

- Anyone can build an island for Fortnite
- Artists and hobbyists, not version control experts
- A repository per project, no setup
- Teams across the world on one island
- In production since launch, 2023
  - The same Lore as the rest of this talk

03 · CREATORS

Two buttons: Sync and Push



Two buttons. The rest is Lore's job.



Exclusive checkout

Two people never edit the same asset



A slice of the project

Only the files you work on, out of terabytes



Never lose work

Every push is a version, every version comes back

Ask the artist to know as little about revision control as possible.

Not a dead end for engineers

CREATORS

Sync · Push
exclusive checkout
no branches to think about

Same repository
one history, the same revisions



nothing exported
nothing bridged
nothing synchronised

ENGINEERS

Branch · merge
review · CI · command line
the full toolset

Simple for the artist. Complete for the engineer. One history.

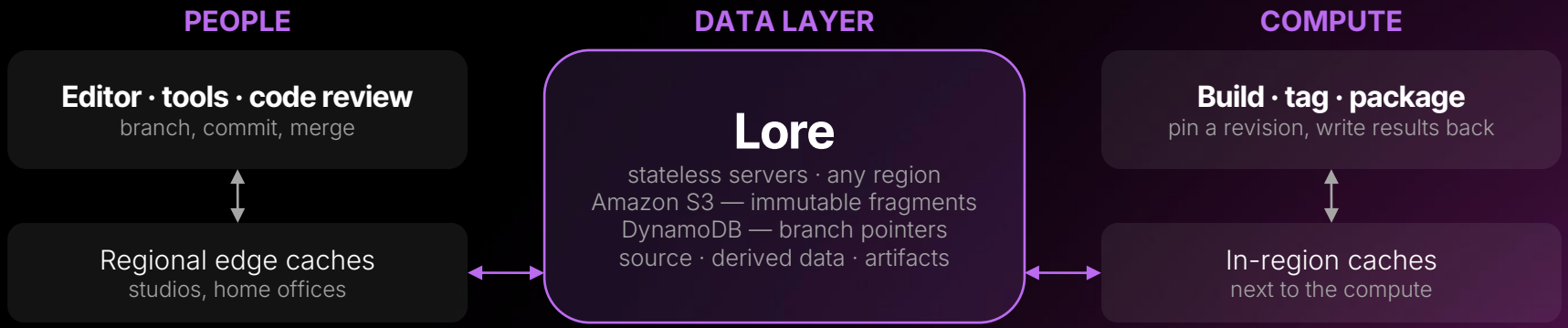
04

Cloud

The same store, under the build pipeline on AWS.

04 · CLOUD

People and jobs on the same data layer



Same store, same identity, same address — whether a person or a job produced it.

One store for the whole pipeline

Source assets by content	→	stored once,
Intermediates their inputs	→	keyed by
Processed output source	→	next to its
Between stages changed moves	→	only what

04 · CLOUD

Data in a content pipeline

The store is the integration point, not the process tree.



Derived data, keyed by its source

Outputs live under the hash of their inputs. Computed once, shared by everyone, never stale.



Metadata on everything

Tags, thumbnails, dependencies, external ids. Typed key-value, findable by value.



Notifications on change

Every new revision is an event. Jobs run the moment content lands, on that exact revision.



Every version, addressable forever

Nothing is overwritten. Pin a revision and reproduce its result a year later, byte for byte.



04 · CLOUD

Content addressed, so data lives next to the compute

- Any copy is as good as the original
- Cache on the studio LAN or in the cloud
- Fetched once, served locally
- No workspace to manage or sync
- Spin one up, do the work, throw it away
  - That is a content pipeline step

Coming this year: a virtual file system

New workspace ready instantly	→	any size,
Local cache at once	→	data served
Prefetching the app defines	→	in the order
New machine time	→	zero startup

04 · CLOUD

What it runs on



Everything above S3 and DynamoDB is a cache. Lose a server, lose nothing.

05

Your studio

Engine-agnostic, open source, in your own AWS account.

05 · YOUR STUDIO

Where to start

A FIFTEEN-PERSON TEAM

Clone, sync, push. A cache when the office grows

A FIVE-HUNDRED-PERSON STUDIO

One pipeline stage first, beside what you run today

ON PERFORCE OR GIT TODAY

Nothing to migrate on day one. Lore runs alongside

ANY ENGINE

Files and revisions, not engine formats

IN CLOSING

Three things to take away

01 Two buttons for artists, the full toolset for engineers.

02 One store, source to cooked output, nothing stored twice.

03 Open source, any engine, your own AWS account.

Thank you

lore.org

Lore is open source, MIT licensed.
Lore is developed in Lore.