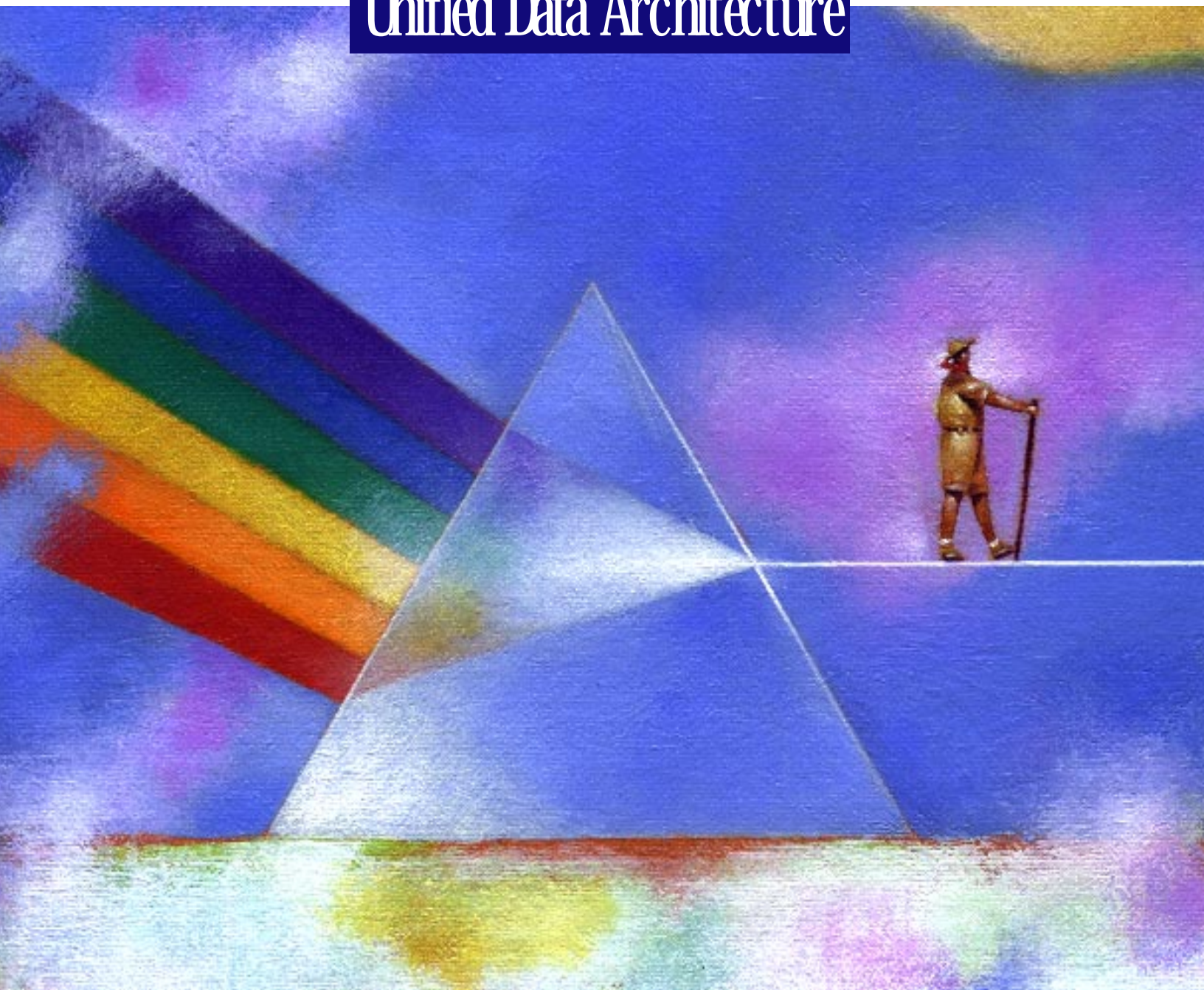


# Caché Components

## Unified Data Architecture



**In their efforts to resolve the inherent "impedance mismatch" between relational and object-oriented representations of data, developers often resort to object-relational mapping. Not only does mapping impede rapid application development, it often exacts a run time performance penalty as well. Caché's Unified Data Architecture eliminates the need for mapping by enabling shared object and relational access to the multidimensional data engine.**

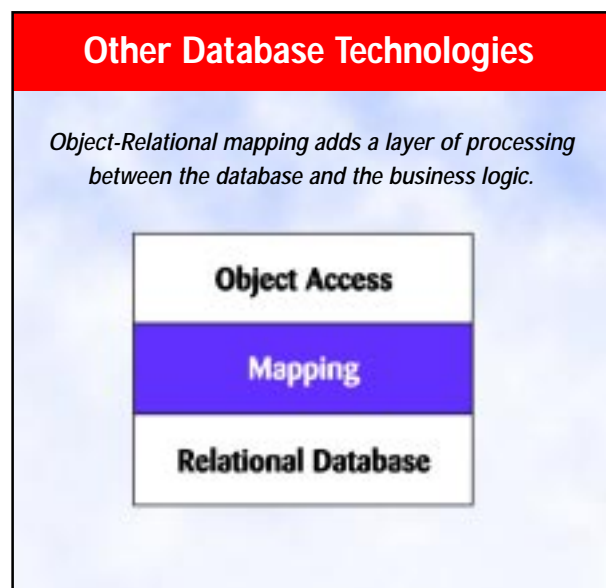
## The Problem: Impedance Mismatch

Today's developers have a problem. On the one hand, they want to use object-oriented development technologies like Java, C++, and COM, because of the superior productivity and richer data models that these environments provide. But on the other hand, their applications often have to access data stored in relational databases or interact with data analysis and reporting tools that use SQL. The problem (often called impedance mismatch) is that relational data is not a good fit for object-oriented languages.

However, "pure" object databases are not a good fit for popular data analysis and reporting tools that rely on SQL. Today's developers need to have both object and relational access to their data.

In order to achieve this goal, developers are often forced to adopt some sort of "object-relational mapping." Whether done by hand or with the aid of a tool, mapping is a tedious process that can significantly slow application development. Even worse, most mapping technology does not allow for data model evolution – the map must be recreated every time the data structures used by an application have changed.

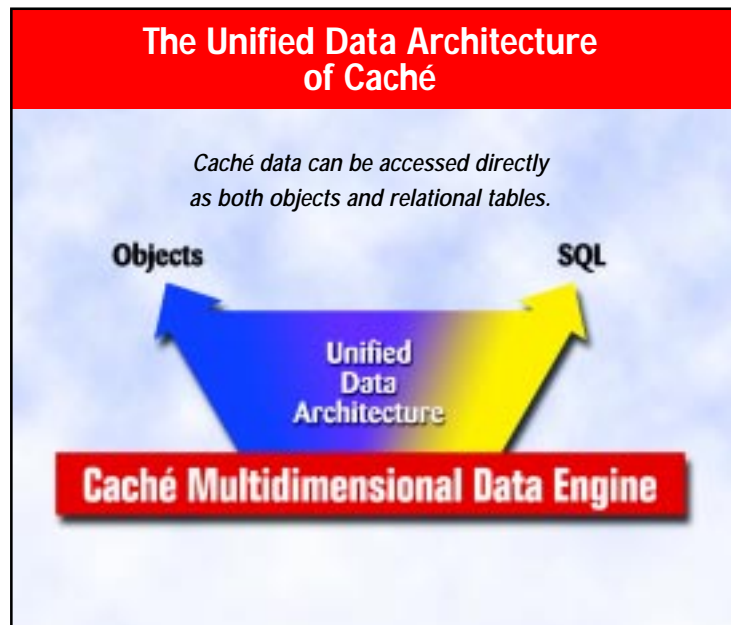
Object-relational mapping may also slow application performance because of the processing overhead required to convert data between object and relational formats.



## The Elegant Solution: Unified Data Architecture

---

Caché's Unified Data Architecture is an elegant solution to the problem of impedance mismatch. That's because Caché's multidimensional data structures are a natural way to store the rich data types characteristic of object-oriented technologies. In addition, it is easy to project multidimensional structures as two-dimensional structures (ie: tables) for access via SQL. Thus, both object and relational data models can simultaneously share Caché's multidimensional data without the need for mapping between formats.



With the Unified Data Architecture, every data structure has a single definition that functions both as an object class and as relational tables. Developers can define Caché object classes (using the Caché Studio or an object design tool) that will be instantly available as tables. And conversely, relational table definitions can be imported to Caché using DDL, and those tables will immediately be usable as objects.

Caché's unified data definitions are automatically created at compile time, and can evolve as data structures and schemas change. Developers never have to map between object and relational representations of data, nor do they have to worry about synchronizing separate data definitions.

**InterSystems**  
**World Headquarters**

---

One Memorial Drive  
Cambridge, MA 02142  
USA  
Phone: +1.617.621.0600  
Fax: +1.617.494.1631

**InterSystems.com**

