DECADE
Internet-Draft
Intended status: Informational
Expires: April 28, 2011

R. Alimi
Google
Y. Yang
Yale University
A. Rahman
InterDigital Communications, LLC
D. Kutscher
NEC
L. Chen
H. Liu
Yale University
October 25, 2010

# DECADE Architecture
draft-alimi-decade-arch-01

## Abstract

Peer-to-peer (P2P) applications have become widely used on the Internet today and make up a large portion of the traffic in many networks.  One technique to improve the network efficiency of P2P applications is to introduce storage capabilities within the network. The DECADE Working Group has been formed with the goal of developing an architecture to provide this capability.  This document presents an architecture, discusses the underlying principles and identifies core components and protocols supporting the architecture.

## Status of this Memo

Copyright Notice

Table of Contents

# 1. Introduction

Peer-to-peer (P2P) applications have become widely used on the Internet today to distribute contents, and they contribute a large portion of the traffic in many networks. The DECADE Working Group has been formed with the goal of developing an architecture to introduce in-network storage to be used by such applications, to achieve more efficient content distribution. Specifically, in many subscriber networks, it is typically more expensive to upgrade network equipment in the "last-mile", because it can involve replacing equipment and upgrading wiring at individual homes, businesses, and devices such as DSLAMs and CMTSs. Thus, it can be cheaper to upgrade core infrastructure involving fewer components that are shared by many subscribers. See [I-D.ietf-decade-problem-statement] for a more complete discussion of the problem domain and general discussion of the capabilities to be provided by DECADE.

This document presents a potential architecture of providing in-network storage that can be integrated into content distribution applications. The primary focus is P2P-based content distribution, but the architecture may be useful to other applications with similar characteristics and requirements. In particular, content distribution applications that may split data into smaller pieces for distribution may be able to utilize DECADE.

The design philosophy of the DECADE architecture is to provide only the core functionality that is needed for applications to make use of in-network storage. With such core functionality, the protocol may be simple and easier to support by storage providers. If more complex functionality is needed by a certain application or class of applications, it may be layered on top of the DECADE protocol.

The DECADE protocol will leverage existing transport and application layer protocols and will be designed to work with a small set of alternative IETF protocols.

This document proceeds in two steps. First, it details the core architectural principles that can guide the DECADE design. Next, given these core principles, this document presents the core components of the DECADE architecture and identifies usage of existing protocols and where there is a need for new protocol development.

This document will be updated to track the progress of the DECADE survey [I-D.ietf-decade-survey] and requirements [I-D.gu-decade-reqs] drafts.

## 2. Entities

### 2.1. DECADE Storage Servers

DECADE storage servers are operated by DECADE storage providers and provide the DECADE functionality as specified in this memo, including mechanisms to store, retrieve and manage data. A storage provider may typically operate multiple storage servers.

### 2.2. DECADE Storage Provider

A DECADE in-storage provider deploys and/or manages DECADE servers within a network. A storage provider may also own or manage the network in which the DECADE servers are deployed.

A DECADE storage provider, possibly in cooperation with one or more network providers, determines deployment locations for DECADE servers and determines the available resources for each.

### 2.3. DECADE Content Providers

DECADE content providers access DECADE storage servers (by way of a DECADE client) to upload and manage data. A content provider can access one or more storage servers. A content provider may be a single process or a distributed application (e.g., in a P2P scenario).

### 2.4. DECADE Content Consumers

DECADE content consumers access storage servers (by way of a DECADE client) to download data that has previously been stored by a content provider. A content consumer can access one or more storage servers. A content consumer may be a single process or a distributed application (e.g., in a P2P scenario). An instance of a distributed application, such as a P2P application, may both provide content to and consume content from DECADE storage servers.

### 2.5. Content Distribution Application

A content distribution application is a distributed application designed for dissemination of possibly-large data to multiple consumers. Content Distribution Applications typically divide content into smaller immutable blocks for dissemination.

The term Application Developer refers to the developer of a particular Content Distribution Application.

### 2.6. Application End-Point

An Application End-Point is an instance of a Content Distribution Application that makes use of DECADE server(s). A particular Application End-Point may be a DECADE Content Provider, a DECADE Content Consumer, or both.

An Application End-Point need not be an active member of a "swarm" to interact with the DECADE storage system. That is, an End-Point may interact with the DECADE storage servers as an offline activity.

## 3. Architectural Principles

We identify the following key principles.

### 3.1. Decoupled Control and Data Planes

The DECADE infrastructure is intended to support multiple content distribution applications. A complete content distribution application implements a set of control functions including content search, indexing and collection, access control, ad insertion, replication, request routing, and QoS scheduling. Different content distribution applications can have unique considerations designing the control and signaling functions. For example, a major competitive advantage of many successful P2P systems is their substantial expertise in how to most efficiently utilize peer and infrastructural resources. Many live P2P systems have their specific algorithms in selecting the peers that behave as the more stable, higher-bandwidth sources. They continue to fine-tune such algorithms. In other words, in-network storage should export basic mechanisms and allow as much flexibility as possible to the control planes to implement specific policies. This conforms to the end-to-end systems principle and allows innovation and satisfaction of specific business goals.

Specifically, in the DECADE architecture, the control plane focuses on the application-specific, complex, and/or processing intensive functions while the data plane provides storage and data transport functions.

- Control plane: Signals details of where the data is to be downloaded from. Also signals the time, quality of service, and receiver of the download. It also provides higher layer meta-data management functions such as defining the sequence of data blocks forming a higher layer content object. These are behaviors designed and implemented by the Application. By Application, we mean the broad sense that include other control plane protocols.

- o Data plane: Stores and transfers data as instructed by the Application’s Control Plane.

Decoupling control plane and data plane is not new. For example, OpenFlow is an implementation of this principle for Internet routing, where the computation of the forwarding table and the application of the forwarding table are separated. Google File System applies the principle to file system design, by utilizing the Master to handle the meta-data management, and the chunk servers to handle the data plane (i.e., read and write of chunks of data). NFS4 also implements this principle.

Note that applications may have different Data Plane implementations in order to support particular requirements (e.g., low latency). In order to provide interoperability, the DECADE architecture does not intend to enable arbitrary data transport protocols. However, the architecture may allow for multiple data transport protocols to be used.

Also note that although an application’s existing control plane functions remain implemented within the application, the particular implementation may need to be adjusted to support DECADE.

## 3.2. Immutable Data Objects

A property of bulk contents to be distributed is that they typically are immutable -- once a piece of content is generated, it is typically not modified. It is not common that bulk contents such as video frames and images need to be modified after distribution.

Many content distribution applications divide content objects into blocks for two reasons: (1) multipath: different blocks may be fetched from different content sources in parallel, and (2) faster recovery and verification: individual blocks may be recovered and verified. Typically, applications use a block size larger than a single packet in order to reduce control overhead.

Common applications whose content matches this model include P2P streaming (live and video-on-demand) and P2P file-sharing content. However, other types of applications may additionally match this model.

DECADE adopts a design in which immutable data objects may be stored at a storage server. Applications may consider existing blocks as DECADE data objects, or they may adjust block sizes before storing in a DECADE server.

Focusing on immutable data blocks in the data plane can substantially simplify the data plane design, since consistency requirements can be relaxed. It also allows effective reuse of data blocks and de-duplication of redundant data.

Depending on specific application requirements, data objects can be complete self-contained resources (such as video files) or chunks of such resources. The DECADE architecture and protocols are agnostic to the nature of the data objects and do not specify a fixed size for them.

Note that immutable content may still be deleted. Also note that immutable data blocks do not imply that contents cannot be modified. For example, a meta-data management function of the control plane may associate a name with a sequence of immutable blocks. If one of the blocks is modified, the meta-data management function changes the mapping of the name to a new sequence of immutable blocks.

### 3.3. Data Object Identifiers

Objects that are stored in a DECADE storage server can be accessed by DECADE content consumers by a resource identifier that has been assigned within a certain application context.

Because a DECADE content consumer can access more than one storage server within a single application context, a data object that is replicated across different storage servers managed by a DECADE storage provider, can be accessed by a single identifier.

Note that since data objects are immutable, it is possible to support persistent identifiers for data objects.

### 3.4. Explicit Control

To support the functions of an application’s control plane, applications must be able to know and control which data is stored at particular locations. Thus, in contrast with content caches, applications are given explicit control over the placement (selection of a DECADE server), deletion (or expiration policy), and access control for stored data.

Consider deletion/expiration policy as a simple example. Applications may require a DECADE server to store content for a relatively short period of time (e.g. for live-streaming data) or may need to store content long term (e.g., for video-on-demand).

## 3.5. Resource and Data Access Control through User Delegation

DECADE provides a shared infrastructure to be used by multiple tenants of multiple content distribution applications. Thus, it needs to provide both resource and data access control.

### 3.5.1. Resource Allocation

There are two primary interacting entities in the DECADE architecture. First, Storage Providers control where DECADE storage servers are provisioned and their total available resources. Second, Applications control data transfers amongst available DECADE servers and between DECADE servers and end-points. A form of isolation is required to enable concurrently-running Applications to each explicitly manage their own content and share of resources at the available servers.

Management of the resources at a server are delegated by a Storage Provider to one or more applications. Applications are able to explicitly and independently manage their own share of resources.

### 3.5.2. User Delegations

Storage providers have the ability to explicitly manage the entities allowed to utilize the resources at a DECADE server. This capability is needed for reasons such as capacity-planning and legal considerations in certain deployment scenarios.

To provide a scalable way to manage applications granted resources at a DECADE server, a layer of indirection is added. Instead of granting resources to an application, the DECADE server grants a share of the resources to a user. The user may in turn share the granted resources amongst multiple applications. The share of resources granted by a storage provider is called a User Delegation.

A User Delegation may be granted to an end-user (e.g., an ISP subscriber), a Content Provider, or an Application Provider. A particular instance of an application may make use of the storage resources:

- granted to the end-user (with the end-user’s permission),
- granted to the Content Provider (with the Content Provider’s permission>, and/or
- granted to the Application Provider.

## 4. System Components

The current version of the document has primarily focused on the architectural principles. The detailed system components will be discussed in the next document revision.

This section presents an overview of the components in the DECADE architecture.

```
           .--------------------------------------------------------------.
           | Application End-Point                                        |
           | .------------.          .-----------------. .----------.     |
           | | App-Layer  |   ...    | App Data Object | | App Data |     |
           | | Algorithms |          |    Sequencing   | |  Naming  |     |
           | '------------'          '-----------------' '----------'     |
           |                                                              |
           | .----------------------------------------------------------. |
           | | DECADE Client                                            | |
           | |                                                          | |
           | | .-------------------------. .--------------------------. | |
           | | | Resource Controller     | | Data Controller          | | |
           | | | .--------. .----------. | | .------------. .-------. | | |
 Native    | | | |  Data  | | Resource | | | |    Data    | | Data  | | | |
  App      | | | | Access | | Sharing  | | | | Scheduling | | Index | | | |
Protocol(s)| | | | Policy | |  Policy  | | | |            | |       | | | |
      .--> | | | '--------' '----------' | | '------------' '-------' | | |
      |    | | '-------------------------' '--------------------------' | |
      |    | |             |                   ^                        | |
      |    | '------------ | ----------------- | -----------------------' |
      |    '-------------- | ----------------- | -------------------------'
      |                    |                   |
      v                    | DECADE            | Standard
.-------------.            | Resource          | Data
| Application |            | Protocol (DRP)    | Transport (SDT)
|  End-Point  |            |                   |
'-------------'            |                   |      Content Distribution
  ^       ^                |                   |          Application
= | ===== | ============== | ================= | ==========================
  |       |                |                   |        DECADE Server(s)
  |       |                |                   |
  |       |             .- | ----------------- | ----------------------.
  |       |             |  |                   v                       |
  |       |             |  |      .----------------.                   |
  |       |             |  |----> | Access Control | <--------.        |
  | DRP   | SDT         |  |      '----------------'          |        |
  |       |             |  |                   ^              |        |
  |       |             |  |                   v              |        |
  |       |             |  |   .---------------------.        |        |
  |       |             |  '-> | Resource Scheduling | <------|        |
  v       v      DRP    |      '---------------------'        |        |
.------------. <------> |                      ^              |        |
|   DECADE   |          |                      v        .------------. |
|   Server   |   SDT    |        .-----------------.    |    User    | |
'------------' <------> |        |    Data Store   |    | Delegation | |
                        |        '-----------------'    | Management | |
                        | DECADE Server                 '------------' |
                        '----------------------------------------------'
```

Figure 1: DECADE Architecture Components

A component diagram of the DECADE architecture is displayed in Figure 1. The diagram illustrates the major components of a Content Distribution Application related to DECADE, as well as the functional components of a DECADE Server.

To keep the scope narrow, we only discuss the primary components related to protocol development. Particular deployments may require additional components (e.g., monitoring and accounting at a DECADE server), but they are intentionally omitted from the current version of this document.

### 4.1. Content Distribution Application

Content Distribution Applications have many functional components. For example, many P2P applications have components to manage overlay topology management, piece selection, etc. In supporting DECADE, it may be advantageous to consider DECADE within some of these components. However, in this architecture document, we focus on the components directly employed to support DECADE.

#### 4.1.1. Data Sequencing and Naming

DECADE is primarily designed to support applications that can divide distributed contents into immutable data objects. To accomplish this, applications include a component responsible for re-assembling data objects and also creating the individual data objects. We call this component Application Data Sequencing. The specific implementation is entirely decided by the application.

In assembling or producing the data objects, an important consideration is the naming of these objects. We call the component responsible for assigning and interpreting application-layer names the Application Data Naming component. The specific implementation is entirely decided by the application.

#### 4.1.2. Native Protocols

Applications may still use existing protocols. Existing protocols used primarily for control/signaling needed by the application, but may also serve as a data transport as they do today; it is important that applications still be designed to be robust (e.g., if DECADE servers are unavailable).

### 4.1.3. DECADE Client

An application may be modified to support DECADE. We call the layer providing the DECADE support to an application the DECADE Client. It is important to note that a DECADE Client need not be embedded into an application. It could be implemented alone, or could be integrated in other entities such as network devices themselves.

#### 4.1.3.1. Resource Controller

Applications may have different Resource sharing policies and Data access policies to control their resource and data in DECADE servers. These policies can be existing policies of applications (e.g., tit-for-tat) or custom policies adapted for DECADE. The specific implementation is decided by the application.

#### 4.1.3.2. Data Controller

DECADE is designed to decouple the control and the data transport of applications. Data transport between applications and DECADE servers uses standard data transport protocols. It may need to schedule the data being transferred according to network conditions, available DECADE Servers, and/or available DECADE Server resources. An index indicates data available at remote DECADE servers. The index (or a subset of it) may be advertised to other Application End-Points.

## 4.2. DECADE Server

DECADE server is an important functional component of DECADE. It stores data from Application End-Points, and provides control and access of those data to Application End-Points. Note that a DECADE server is not necessarily a single physical machine, it could also be implemented as a cluster of machines.

### 4.2.1. Access Control

An Application End-Point can access its own data or other Application End-Point’s data (provided sufficient authorization) in DECADE servers. Application End-Points may also authorize other End-Points to store data. If an access is authorized by an Application End-Point, the DECADE Server will provide access.

Note that even if an request is authorized, it may still fail to complete due to insufficient resources by either the requesting Application End-Point or the providing Application End-Point.

### 4.2.2. Resource Scheduling

Applications may apply their existing resource sharing policies or use a custom policy for DECADE. DECADE servers perform resource scheduling according to the resource sharing policies indicated by Application End-Points as well as configured User Delegations.

Access control and resource control are separated in DECADE server. It is possible that an Application End-Point provides only access to its data without any resources. In order to access this data, another Application End-Point may use the granted access along with its own available resources to store or retrieve data from a DECADE Server.

### 4.2.3. Data Store

Data from applications may be stored into disks and explicitly or automatically (e.g., after a TTL) deleted from disks. It may be possible to perform optimizations in certain cases, such as avoiding writing temporary data (e.g., live streaming) to disk.

## 4.3. Protocols

The DECADE Architecture uses two protocols. First, the DECADE Resource Protocol is responsible for communicating access control and resource scheduling policies to the DECADE Server. Second, standard data transport protocols (e.g., WebDAV or NFS) are used to transfer data objects to and from a DECADE Server. The DECADE architecture will specify a small number of Standard Data Transport instances.

Decoupling the protocols in this way allows DECADE to both directly utilize existing standard data transports and to evolve independently.

It is also important to note that the two protocols do not need to be separate on the wire. For example, the DECADE Resource Protocol messages may be piggybacked within extension fields provided by certain data transport protocols. However, this document considers them as two separate functional components for clarity.

### 4.3.1. DECADE Resource Protocol

The DECADE Resource Protocol is responsible for communicating both access control and resource sharing policies to DECADE Servers used for data transport.

The DECADE architecture specification will provide exactly one DECADE Resource Protocol.

### 4.3.2. Standard Data Transports

Existing data transport protocols are used to read and write data from a DECADE Server. Protocols under consideration are WebDAV and NFS.

## 4.4. DECADE Data Sequencing and Naming

We have discussed above that an Application may have its own behavior for both sequencing and naming data objects. In order to provide a simple and generic interface, the DECADE Server is only responsible for storing and retrieving individual data objects.

The issue of naming data objects at the DECADE server would benefit from additional feedback. There are multiple options that have been considered:

- Self-certifying Name: The name of a data object may be a hash of its contents. Advantages of this scheme include simplicity and low probability of naming collisions without requiring any identifiers or namespaces to be allocated. Disadvantages of this scheme include collision in identifiers (with low probability) and introduction of an additional distribution delay due to the necessity of reading the full object to compute its hash before advertising its availability.

- Application-specified Name: The name of a data object is specified by the application itself. For example, this could be a function of the application’s content identifier and index of the data object. To avoid conflicts, identifiers could be assigned to particular applications. An advantage of this scheme is that collisions could be avoided. A disadvantage is that assigning identifiers introduces additional management complexity.

- Server-specified Name: The name of a data object is specified by the DECADE server upon initially being stored. An advantage of this approach is that naming conflicts can be completely avoided without requiring particular identifiers to be assigned to applications. A disadvantage is that it introduces additional latency between the time when a application may upload a data object and advertise availability of the data object at the DECADE Server.

The current preferred design is to use self-certifying names. However, additional feedback is welcomed.

### 4.5. In-Network Storage Components Mapped to DECADE Architecture

In this section we evaluate how the basic components of an in-network storage system identified in Section 3 of [I-D.ietf-decade-survey] map into the DECADE architecture.

It is important to note that complex and/or application-specific behavior is delegated to applications instead of tuning the storage system wherever possible.

#### 4.5.1. Data Access Interface

Users can read and write objects of arbitrary size through the DECADE Client's Data Controller, making use of a standard data transport.

#### 4.5.2. Data Management Operations

Users can move or delete previously stored objects via the DECADE Client's Data Controller, making use of a standard data transport.

#### 4.5.3. Data Search Capability

Users can enumerate or search contents of DECADE servers to find objects matching desired criteria through services provided by the Content Distribution Application (e.g., buffer-map exchanges, a DHT, or peer-exchange). In doing so, End-Points may consult their local data index in the DECADE Client's Data Controller.

#### 4.5.4. Access Control Authorization

All methods of access control are supported: public-unrestricted, public-restricted and private. Access Control Policies are generated by a Content Distribution Application and provided to the DECADE Client's Resource Controller. The DECADE Server is responsible for implementing the access control checks.

#### 4.5.5. Resource Control Interface

Users can manage the resources (e.g. bandwidth) on the DECADE server that can be used by other Application End-Points. Resource Sharing Policies are generated by a Content Distribution Application and provided to the DECADE Client's Resource Controller. The DECADE Server is responsible for implementing the resource sharing policies.

#### 4.5.6. Discovery Mechanism

This is outside the scope of the DECADE architecture. However, it is expected that DNS or some other well known protocol will be used for the users to discover the DECADE servers.

#### 4.5.7. Storage Mode

DECADE Servers provide an object-based storage mode. Immutable data objects may be stored at a DECADE server. Applications may consider existing blocks as DECADE data objects, or they may adjust block sizes before storing in a DECADE server.

## 5. Security Considerations

This document currently does not contain any security considerations beyond those mentioned in [I-D.ietf-decade-problem-statement].

## 6. IANA Considerations

This document does not have any IANA considerations.

## 7. Informative References

[I-D.ietf-decade-problem-statement]
Yongchao, S., Zong, N., Yang, Y., and R. Alimi, "DECoupled Application Data Enroute (DECADE) Problem Statement", draft-ietf-decade-problem-statement-00 (work in progress), August 2010.

[I-D.ietf-decade-survey]
Alimi, R., Rahman, A., and Y. Yang, "A Survey of In-network Storage Systems", draft-ietf-decade-survey-01 (work in progress), October 2010.

[I-D.gu-decade-reqs]
Yingjie, G., Bryan, D., Yang, Y., and R. Alimi, "DECADE Requirements", draft-gu-decade-reqs-05 (work in progress), July 2010.

## Authors’ Addresses

Richard Alimi
Google

Email: ralimi@google.com

Y. Richard Yang
Yale University

Email: yry@cs.yale.edu

Akbar Rahman
InterDigital Communications, LLC

Email: akbar.rahman@interdigital.com

Dirk Kutscher
NEC

Email: dirk.kutscher@neclab.eu

Lijiang Chen
Yale University

Email: lijiang.chen@yale.edu

Hongqiang Liu
Yale University

Email: hongqiang.liu@yale.edu