

# Snapshot on the EDOS Distribution System

Serge Abiteboul  
INRIA  
serge.abiteboul@inria.fr

Itay Dar  
Tel Aviv University  
itay.dar@post.tau.ac.il

Radu Pop  
INRIA-Mandriva  
radu.pop@inria.fr

Gabriel Vasile  
INRIA  
gabriel.vasile@inria.fr

Dan Vodislav  
CNAM  
vodislav@cnam.fr

February 12, 2007

## 1 Introduction

The open-source software communities currently face an increasing complexity of managing the software content among their developers and contributors. This is mainly due to the continuously growing size of the software, the high frequency of the updates, and the heterogeneity of the participants. We propose a distribution system that tackles two main issues in the software content management: efficient content dissemination through a P2P system architecture, and advanced information system capabilities, using a distributed index for resource location. We believe that both aspects are essential in improving the performances of the general system and each particular choice in the system's architecture completes the overall functionalities. The aim of the EDOS [7] Distribution System is to merge and to exploit the proved efficiency of some existing subsystems.

Due to space limitations, the paper only presents an overview of the EDOS distribution system. More details on the system can be found in [9]. The paper is organized as follows: section 2 describes the functionalities of the system, we give a brief overview on the software architecture and system implementation in section 3, and we conclude in section 4.

## 2 System functionalities

The goal of the EDOS distribution system is to efficiently disseminate open-source software (referred at a more general level as *data* or *content*) through

the Internet. Published by a main server, data is disseminated in the network to other computers (mirrors, end users), that get copies of the published content.

EDOS system is articulated around a distributed, P2P information system that stores and indexes *content metadata*. This metadata-based information system allows querying and locating data in the EDOS network.

The choices for the functional architecture are driven by the three main aspects that define the system: the data model used for the content management, the actors in the system, and their roles in the P2P architecture.

## 2.1 Data model

The basic **data unit** in the EDOS distribution system is the *package*, represented in most Linux distributions by an RPM file, that contains source and binary code in a compressed form.

There are three types of data units that may be published and distributed in the EDOS system:

- *Packages*, the main data unit type, containing code and binaries in an RPM file.
- *Utilities*, represented by individual files used in the software installation process.
- *Collections*, that group together packages, utilities or sub-collections, to form a hierarchical organization of data.

Figure 1 presents an example of data units published in EDOS. We adopted the hierarchical organization of data in the Mandriva Cooker distribution, where packages and utility files may be grouped in collections at several levels, providing various levels of data granularity.

**Metadata management** is a key issue in the distribution process. We aim at building a global, distributed information system about data to be disseminated in the network. This system is fed with content metadata. The ability of expressing complex queries over metadata and to provide effective distributed query processing is a major contribution of this project.

In the largest sense, metadata consists of the set of properties that characterize data units. For each data unit type, the set of metadata properties is different. In the EDOS data model, metadata includes:

- for *packages*: name, version, size, checksum, license, dependencies to other packages, etc. In the Mandriva Linux distribution, this informa-

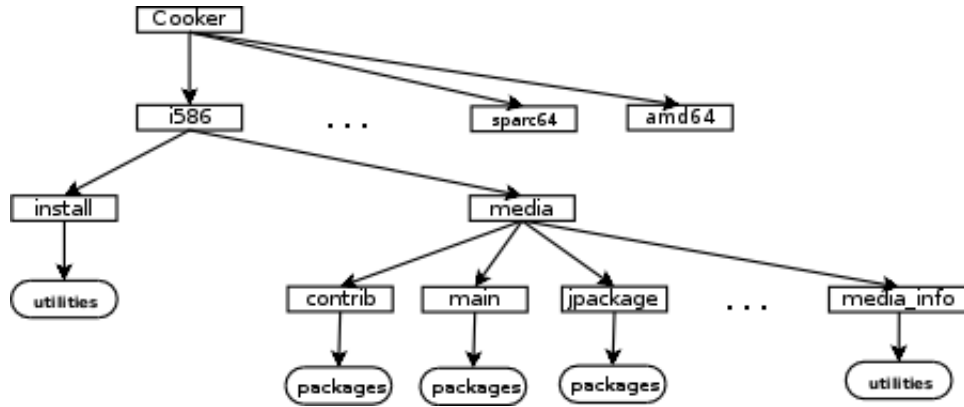


Figure 1: Data units for EDOS distribution

tion can be extracted from the RPM files (generally, a large number of metadata properties).

- for *utilities*: name, version, size, etc.
- for *collections*: name, version, collection composition, etc.

In order to be published in the EDOS distribution system, metadata is represented in the form of XML files. Data dissemination is initiated by *publishing* data units in the system. More precise, the publish action consists in generating the metadata characterizing the data units and indexing it in the distributed index.

## 2.2 Actors and roles in the P2P architecture

The EDOS distribution system has a peer-to-peer (P2P) architecture, including all the computers that participate to the dissemination process over the Internet.

Peers of the EDOS P2P distribution system may be classified in *three categories of actors*, similar to those existing in classical content distribution systems:

- **Publisher**, corresponds to the main distribution server, that introduces new content in the EDOS distribution system.
- **Mirrors**, correspond to secondary servers, trusted peers that keep copies of the published content and provide additional downloading sources in the distribution network.

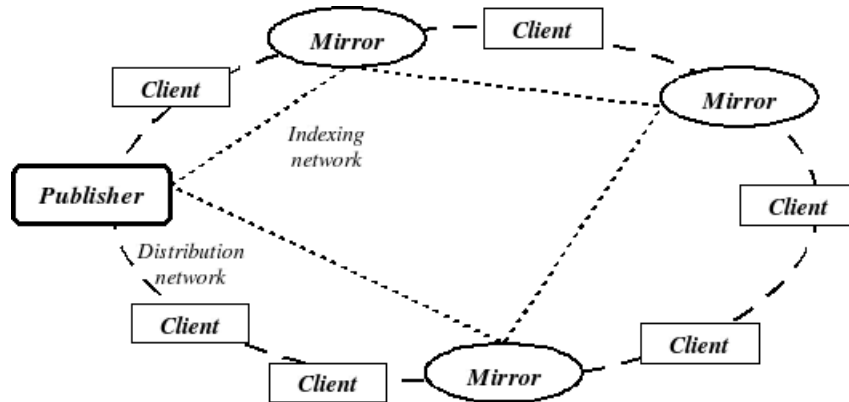


Figure 2: Actors in EDOS P2P content distribution

- **Clients**, correspond to end user computers, that download content from mirrors. Clients are not trusted for participating to index management. They need an entry in the indexing network in order to have access to querying metadata and to publishing replicas.

Figure 2 presents the P2P distribution network with the three kinds of actors. The EDOS P2P distribution system comes with new interaction between actors, as a consequence of the P2P architecture, and with new actor roles, as a consequence of the new system functionalities.

The P2P network structure is influenced by the data management policy in the EDOS distribution system, which handles information at two levels: *data/content*, copied from peer to another, and *metadata*, indexed for query processing. As presented in Figure 2, actors are connected in *two distinct networks*:

- *The distribution network*, containing all the peers that store and share EDOS data, i.e. software packages, utilities and collections.
- *The indexing network*, storing the index on content metadata, distributed among the indexing network peers. Because of security reasons, we choose to organize the indexing network on a *trusted* subset of peers in the distribution network.

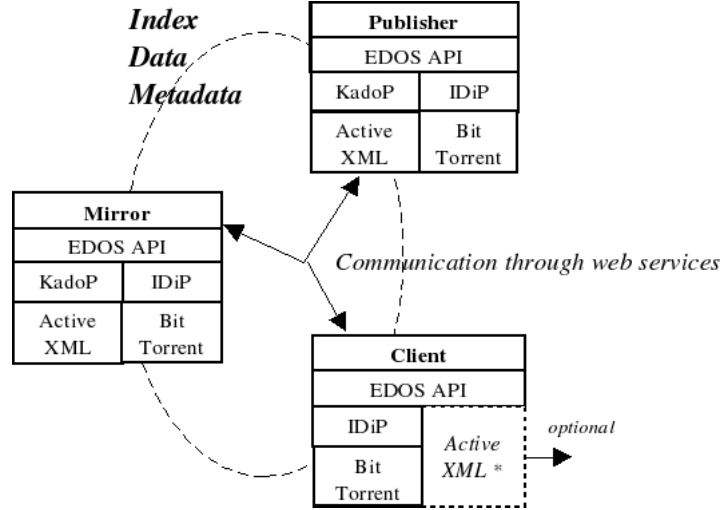


Figure 3: Software modules for EDOS distribution actors

### 3 Architecture and implementation

The functional architecture presented in section 2 is implemented in a real system, whose software architecture is illustrated in Figure 3.

Based on the *EDOS distribution API* outlined in [8], the EDOS distribution platform realizes the functionalities described in section 2 by using a set of software modules: *KadoP*, *ActiveXML*, *IDiP* and *BitTorrent/Azureus*.

- **KadoP** [5, 3]: distributed index for EDOS metadata, that allows publishing and querying of content unit metadata, represented as (Active)XML files
- **ActiveXML** [4, 1]: provides an extended XML format for EDOS metadata, storage for metadata files published in KadoP, and web services for inter-peer communication.
- **IDiP**: dissemination platform that implements functionalities for the flash-crowd usage scenario.
- **BitTorrent** [6, 2]: used for multicast in IDiP, in the flash-crowd dissemination scenario. May also be used for individual download of content units during off-peak periods.

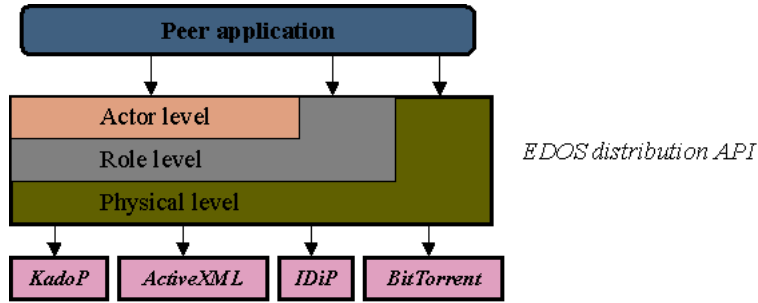


Figure 4: EDOS distribution API structure

### 3.1 EDOS distribution API

The EDOS distribution API is a Java API for writing distribution applications. This API is built on top of the software modules presented above and provides content distribution functionalities at three abstraction levels (see Figure 4).

1. **Physical level:** the lowest level, providing distribution peer basic functionalities. Programming distribution applications at the physical level requires more effort, but offers the greatest flexibility.
2. **Role level:** higher level, providing classes/interfaces for each role in the distribution network, i.e. publishing, downloading, replicating, querying, and subscribing.
3. **Actor level:** highest level, providing classes/interfaces for each actor type (PublisherPeer, MirrorPeer, ClientPeer). An actor plays several roles, e.g. the publisher may publish, manage subscriptions and notifications, be a dissemination server, etc.

### 3.2 Peer application

The prototype of the EDOS distribution system is implemented as a set of web applications on top of the EDOS distribution API. Depending on the actor type (Publisher, Mirror or Client), each peer in the network runs a Java/JSP web application, providing the EDOS distribution functionalities specific to that actor.

- The *Publisher's* main actions are: publishing new content in the distribution system, accepting subscription from the clients, and sending notifications for the updates.

- The *Mirror/Client* can subscribe to communication channels, query the system's index to locate resources, and download content from the system.

These peer applications are written in JSP (Java Server Pages) and need a Tomcat web server for deployment, with Axis module for web service implementation.

## 4 Preliminary tests

The basic test scenario that we put in place consisted in linking together 6 machines from different locations over the Internet. The publisher peer was placed at Mandriva, a mirror and a client peer in INRIA's local network, a second mirror and a second client peer in the local network at University of Paris 7, and finally a third client peer at Caixa Magica. The main issue that we encountered was the network address translation (NAT) used in most of the local networks. The current version of FreePastry (the implementation of the distributed index used in KadoP module) does not support NAT, therefore we were constrained to a limited number of machines using a public IP address. We try to overcome this limitation by considering the STUN (Simple Traversal of UDP through NATs) protocol or NAT traversal using relays.

For passing the tests to a larger scale we used a cluster of around 100 machines at the University of Orsay. The deployment of the applications is done automatically by simple scripts. The peer communication is relatively fast compared to a real case deployment over the Internet, but the advantage of this test framework is the possibility to focus on the indexing performances of the KadoP module.

The publication of a new release is a complex process that involves several intermediate steps. To each data unit we associate an XML metadata file which is parsed and indexed at the KadoP level. Each key-value element of metadata is stored in the distributed index by calling the FreePastry primitives. Therefore, a new connection to the index is required for sending every metadata element. Considering 6000 data units for a new release and an average number of 200 metadata elements per data unit, we experienced very long publication times, up to one hour for indexing all the keys in the distributed index.

The required optimisation focuses on lowering the number of connections to the index and can be achieved by implementing a caching technique for the metadata elements.

## 5 Conclusion and future work

We presented in this paper a P2P architecture for the open-source content dissemination and we described the functional and software architectures of the distribution system. In a preliminary set of tests it turned out that the publication of a new release is an intensive and time consuming process. We intend to explore optimisation of mass publication in KadoP module. Also for a large scale deployment of the system, an alternative framework for peer communication is planned.

## References

- [1] ActiveXML web page. <http://activexml.net>.
- [2] BitTorrent protocol specification. <http://wiki.theory.org/BitTorrentSpecification>.
- [3] KadoP web page. <http://gemo.futurs.inria.fr/projects/KadoP>.
- [4] S. Abiteboul, O. Benjelloun, I. Manolescu, T. Milo, and R. Weber. Active XML: Peer-to-Peer Data and Web Services Integration. In *VLDB*, pages 1087–1090, 2002.
- [5] S. Abiteboul, I. Manolescu, and N. Preda. Constructing and querying peer-to-peer warehouses of XML resources . In V. T. Chris Bussler, editor, *Second International Workshop on Semantic Web and Databases (SWDB)*. Springer-Verlag, 2004.
- [6] B. Cohen. Incentives Build Robustness in BitTorrent. In *Proceedings of the Workshop on Economics of Peer-to-Peer Systems*, Berkeley, CA, USA, 2003.
- [7] EDOS project: Environment for the development and Distribution of Open Source software. <http://www.edos-project.org>.
- [8] WP4. Edos deliverable 4.1: Distribution of code and binaries over the internet, 2005. <http://www.edos-project.org/xwiki/bin/download/Main/D4-1/edos-d4.1.pdf>.
- [9] WP4. Edos deliverable 4.2.2: Report on the p2p dissemination system, 2006. <http://www.edos-project.org/xwiki/bin/download/Main/D4-2-2/edos-d4.2.2.pdf>.