NetarchiveSuite Overview

software to harvest, archive and browse large parts of the internet.

Printer friendly version

Contents

Introduction

The primary function of the NetarchiveSuite is to plan, schedule and archive web harvests of parts of the internet. We use Heritrix as our webcrawler. The NetarchiveSuite can organize three different kinds of harvests:

- Event harvesting (organize harvests of a set of domains related to a specific event (e.g. 9/11, Royal Weddings, Elections and so on)).
- Selective harvesting (recurrent harvests of a set of domains).
- Snapshot harvesting (organizing a complete snapshot of all known domains)

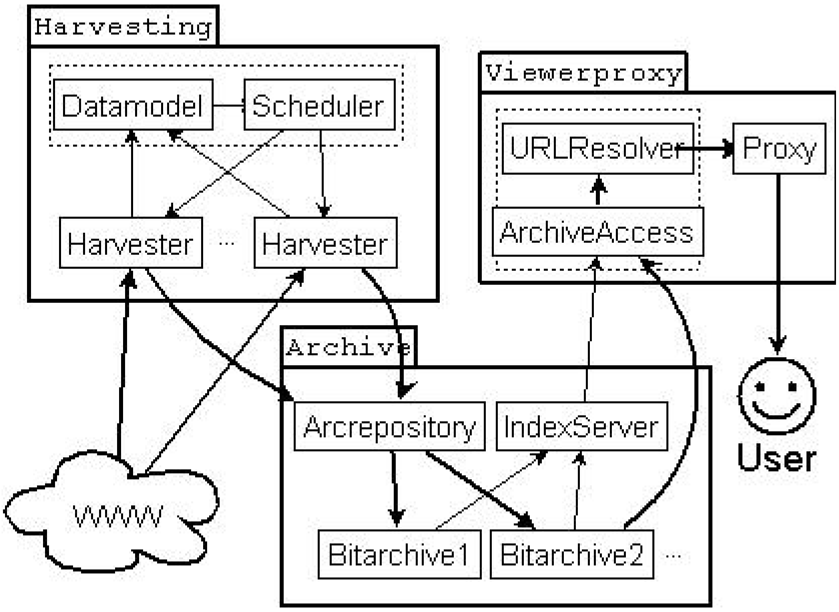
The software has been designed with the following in mind:

- Friendly to non-developers - designed to be usable by librarians and curators with a minimum of technical supervision
- Low maintenance - easy setup of automated harvests, automated bit-integrity checks, and simple curation tools
- High bit-preservation security - replication and active integrity tests of large data contents
- Loosely coupled - the suite consists of modules the can be integrated individually, or be used as one large webarchiving system

The modules in the NetarchiveSuite

The NetarchiveSuite is split into four main modules: One module with common functionality and three modules corresponding to ingesting, archiving and accessing

(See illustration)



dk.netarkivet.common module

This module contains the framework, and utilities used by the whole suite, like exceptions, settings, messaging, filetransfer (RemoteFile), and logging. It also defines the interfaces used to communicate between the different modules, to support alternative implementations.

dk.netarkivet.harvester module

This module handles defining, scheduling, and performing harvests.

- Harvesting uses Heritrix from Internet Archive as the crawler, the harvesting module allows flexible automated definitions of harvests. The system allows the full power of Heritrix, given knowledge of the Heritrix crawler. NetarchiveSuite wraps the crawler in an easy-to-use interface that handles scheduling and configuring the crawl, and distributing it to several crawling servers.
- The harvester module allows for deduplication, using an index from the archive to avoid storing URLs already crawled. This uses the deduplicator module from Kristinn Sigurðsson.
- The harvester module supports packaging metadata about the harvest together with the harvested data.

dk.netarkivet.archive module

This module allows running a repository with replication, active bit consistency checks for bitpreservation, and support for distributed batch jobs on the archive.

- The archiving component offers a secure environment for storing your harvested material. It is designed for high preservation guarantees on bit preservation.
- It allows replication of data on different locations, and distribution of content on several servers on each location. It supports different software and hardware platforms.
- The module allows for distributed batch jobs, running the same jobs on all servers at a location in parallel, and merging the results.
- An index of data in the archive allows fast access to the harvested materials.

dk.netarkivet.viewerproxy module

This module gives access to previously harvested material, through a proxy solution.

- The viewerproxy component supports transparent access to the harvested data, using a proxy solution, and an archive with an index.
- Support for browsing an entire crawl (like a snapshot or event harvest) or a single job (what one machine harvested).
- Allows collecting unharvested URLs while browsing, for use in curation, to include these URLs in the next crawl.

For developers

- The modules are loosely coupled, communicating through interfaces, with the implementation replacable without recompiling.
- A rich number of settings in an XML structure allows for numerous ways of tweaking the applications for special needs.
- All design and code is peer-reviewed.
- There is Javadoc and implementation comments throughout the code.
- The code is tested with unit tests and thorough release tests.
- Development happens in a well-defined development model (originally based on evolutionary prototyping).
- NetarchiveSuite is available under a well-known, integration-friendly, open source license (LGPL).