# Architecture du système POESIA de filtrage de contenu Internet

B. Starynkévitch[1] M.Daoudi[2]

[1] Commissariat à l'Énergie Atomique - DRT/LIST/DTSI/SLA
Bât. 528, CEA/Saclay, 91191 Gif/Yvette cedex, FRANCE

[2] MIIRE Telecom Lille 1
Cité Scientifique - rue G.Marconi - 59659 Villeneuve d'Ascq cedex, FRANCE

basile.starynkevitch@cea.fr daoudi@enic.fr

## Résumé

POESIA *est un logiciel libre (sous licence GPL) de filtrage de contenu Internet, pour protéger la jeunesse de contenus inappropriés. Il est multi-canal (Web, SMTP, NNTP...) et multi-modal (texte, image...). Il est bâti autour d'un moniteur, échangeant avec des sources externes, et des filtres dédiés par un protocole spécifique mais extensible. Ces filtres traitent le langage naturel, les images, les liens hypertextuels statiques et dynamiques. Ils peuvent s'activer par des messages de continuations et envoyer des scores à des mécanismes de décisions qui rejettent ou acceptent in fine le contenu à filtrer. Ces scores sont stockés pour être réutilisés si possible. Le caractère libre de* POESIA *et son architecture adaptative et réutilisable en permet facilement l'extension.*

## Mots clefs

Web, HTTP, SMTP, Filtrage de contenu, logiciel libre, analyse d'images, traitement du language naturel, analyse statique

# 1 Introduction

## 1.1 Contexte et présentation du projet POESIA

Le projet européen POESIA (Public Opensource Environment for a Safer Internet Access) [1] est motivé par la protection de la jeunesse européenne de contenus ou d'usage inappropriés de l'Internet. Il est partiellement financé (IAP2117/27572) par la Commission Européenne dans le cadre du *safer Internet Action Plan* [2]. Afin d'assister le pédagogue en charge d'une classe d'enfants ou de jeunes naviguant sur l'Internet, un système logiciel de filtrage est utile. Un filtre européen de contenu Internet en logiciel libre[1] est préférable à une solution commerciale le plus souvent américaine ; un logiciel libre se diffuse facilement, et la revue du code source par les pairs permet d'en assurer la pérennité, la qualité et la conformité aux usages et valeurs européens.

Le système POESIA s'installe sur un PC *dédié sous Linux* séparant le réseau local des postes[2] à filtrer (classe, bibliothèque, cybercafé) de la connnection Internet (xDSL, campus LAN ...). Il ne peut donc pas être contourné.

POESIA est un *filtre multi-canal* : il filtre aussi bien le Web (protocole HTTP [3, 4]) que le courrier électronique (SMTP, POP3, ...) ou les forums (NNTP). Il peut être étendu à d'autres protocoles. Le système POESIA est *multi-modal* : il filtre aussi bien les images (GIF, PNG, JPEG, ...) que les textes (HTML, XHTML, ou plain) et peut s'étendre à d'autres types de contenus par ajout de filtres spécifiques.

## 1.2 Vue d'ensemble du système POESIA

POESIA est bâti autour d'un programme moniteur (codé en Ocaml [5]). Ce moniteur, en cours de développement, communique avec des sources d'information, des filtres spécifiques -en cascade ou en parallèle-, des mécanismes de décisions. Le moniteur est donc le chef d'orchestre du système.

Des filtres spécifiques sont lancés et reliés au moniteur. POESIA comprend des filtres pour le traitement des images, des filtres pour le traitement du langage naturel (anglais, espagnol, italien), des filtres pour l'analyse de liens, y compris les liens dynamiques par analyse statique de Javascript. Ces techniques s'ajoutent aux techniques ordinaires de filtrage (par mots-clés, par listes positives ou négatives d'URLs, par scores PICS [6]). Chaque filtre produit un score numérique (scalaire ou vectoriel) sur le contenu analysé.

L'extension des domaines filtrés (par exemple le racisme via *Princip* [7]) ou l'ajout de modes ou types de contenu, et de canaux filtrés peut se faire par ajout de filtres ou de sources d'information supplémentaires.

Des mécanismes de décisions (à seuil, à réseaux de neurones, bayésiens, à base de règles...) reçoivent du moniteur les scores des filtres et produisent la décision finale d'accepter ou de rejeter le contenu à filtrer.

[1] Les logiciels développés dans POESIA sont sous licence GNU GPL ou LGPL.

[2] Les postes Internet sont quelconques : PC sous Windows ou Linux, Mac, bornes de navigation, ...

Pour mutualiser le coût du filtrage, les scores calculés sont stockés par le moniteur pour pouvoir être réutilisés si ce même contenu est filtré une seconde fois.

# 2 Les sources d'information et le moniteur

## 2.1 Les sources d'information

Le Web est le contenu le plus important à filtrer. POESIA utilise le protocole ICAP (Internet Content Adaptation Protocol) [8] pour encapsuler les requêtes et les réponses HTTP à filtrer. Les postes utilisateurs utilisent, comme proxy HTTP transparent[3], une version SQUID/ICAP du cache Squid[9, 10, 11] modifiée pour être un client ICAP [12]. Toute requête HTTP des postes clients à filtrer arrive donc sur SQUID/ICAP. Celui-ci est configuré pour la retransmetttre sur le serveur Web d'origine, puis en attendre la réponse. Ensuite, SQUID/ICAP encapsule la réponse HTTP (ainsi que l'URL requise) dans une requête `RESPMOD` (response modification) vers le moniteur POESIA, qui est donc (pour le filtrage du Web) un serveur ICAP. Le moniteur traite cette requête ICAP en utilisant ses filtres et mécanismes de décisions pour produire une réponse ICAP encapsulant soit la réponse HTTP si le contenu est accepté, soit une réponse erreur HTTP 403 Forbidden si le contenu est filtré. Lorsque le filtrage dépasse un certain temps (configurable, dizaine de secondes), le moniteur doit renvoyer une erreur HTTP 408 Request Timeout ou 404 Not found avec un message d'erreur invitant à refaire la requête HTTP initiale plus tard. Dans ce dernier cas, le filtrage continue. La fonction de cache de SQUID/ICAP est conservée sur le contenu.

Pour filtrer le courriel (SMTP, POP3, ...), on configure le MTA exim [13] (pour SMTP) pour qu'il envoie via un filtre système d'EXIM (cf ch.10 de [13]) spécifique à POESIA chaque courrier à filtrer au moniteur. Les clients `fetchmail` et serveurs `teapop` ou `pop3d` POP3 éventuels sont inchangés et utilisent donc EXIM ou son spool `/var/mail`.

Le filtrage des forums (NNTP) se fait de façon analogue, par exemple en modifiant `leafnode` ou un serveur NNTP (`cnews, inn`...) pour dialoguer avec POESIA.

Le moniteur de POESIA récupère donc les contenus (avec leur URLs pour le Web) à filtrer et les transmet aux filtres spécifiques via un protocole interne à POESIA, sur des tubes nommés. Les filtres et les mécanismes de décisions (dont l'ensemble et le nombre sont configurables) sont gérés par le moniteur. Certains filtres ne sont que des transformateurs de contenus (par exemple `dehtml` enlève les balises d'un document HTML) ou même triviaux (par exemple produisant un score constant).

[3]La couche parefeu Netfilter de linux2.4 et le logiciel Squid peuvent être configurés pour rediriger de manière transparente -sans modification des navigateurs clients- le trafic sur le port 80 vers Squid

## 2.2 le moniteur et son protocole interne

Le moniteur lance les filtres spécifiques et les mécanismes de décisions (ou décideurs) et communique avec eux. Les filtres et décideurs ne communiquent directement (en asynchronie) qu'avec le moniteur POESIA. Ils en recoivent des requêtes et émettent soit des scores de filtrage, soit des demandes de continuation, qui permettent d'enchaîner les traitements. Les filtres traitent chacun (consécutivement ou en parallèle) plusieurs requêtes, et sont redemarrés par le moniteur s'ils plantent.

Les messages échangés (requêtes, scores, demandes de continuation) entre le moniteur et ses filtres et ses décideurs ont un format commun (vaguement inspiré de HTTP), essentiellement textuel. La figure 1 est un exemple fictif de message, qui commence par une ligne de commande, contient des attributs simples (l.2) ou composites (l.3-5), et un corps de message (l.7-10) précédé de sa taille (l.6). Les valeurs d'attributs composites ont une syntaxe lispienne (plus concise et plus simple à générer et à lire que du XML) et sont parfois attribuées (l.5 attribut `a` valant le nom `t`) . Le corps du message peut contenir des octets quelconques, par exemple, ceux d'une image JPEG ou PNG) non encodés. Ce format de message est suffisamment générique, facile à générer et à lire.

| **n°** | **ligne** | |
|---|---|---|
| *ligne de commande* (verbe, numéro de requête) : | | |
| 1 | `REQUEST␣657` | |
| *lignes optionnelles d'attributs :* | | |
| 2 | `␣url=␣"http://bbc.com/"` | |
| 3 | `␣id=␣(etags␣"_1234-56")` | |
| 4 | `␣comp=␣(foo␣1␣-3.14)` | |
| 5 | `␣comp2=␣(bar␣(b␣x␣2␣:a␣t))` | |
| *corps (optionnel) du message* | | |
| 6 | `81` | taille des données |
| 7 | `<html><head><title>` | |
| 8 | `exemple</title></head>` | |
| 9 | `<body><h1>bonjour</h1>` | |
| 10 | `</body></html>` | |
| saut de page obligatoire : | | |
| 11 | formfeed | caractère final |

Figure 1 – *Exemple de message*

Les noms (de commandes, d'attributs, de valeurs composites, ...) apparaissant dans un message sont conventionnels entre filtres, ou bien même dans le moniteur.

Une réponse HTTP est dirigée d'abord vers un ensemble de filtres initiaux choisi selon son Content-type. Pour une image, c'est un filtre d'image; pour du HTML, c'est `dehtml` (pour en ôter les balises), l'analyseur d'image dans les pages, l'analyseur de Javascript, etc.... Le corps du message est le contenu à filtrer. Au-delà d'un volume configurable, le moniteur peut écrire ce contenu dans un fichier temporaire (qu'il supprimera quand la requête aura

été traitée) et transmettre (par un attribut `tempfile`) le nom de ce fichier.

Un filtre peut produire un score (par un message `SCORE`) qui est alors stocké par le moniteur. Celui-ci reproduira instantanément ce score si une autre requête de filtrage vers le même filtre demande le même contenu. La base des scores est indexée par des triplets $(filtre, URL, Etag)$ où $Etag$ est l'entête Etag de la réponse HTTP s'il existe, ou autrement par $(filtre, URL, Md5)$ où $Md5$ est la signature md5 du contenu. Les scores sont effacés de la base au bout d'un certain temps configurable (heures ou jours), par des stratégies inspirées de travaux récents : [14, 15, 16, 11].

Un filtre peut aussi demander la continuation (par un message `CONT`) du traitement. Il indique alors au moniteur le nom du (ou des) filtres à activer. Le corps de ce message est alors renvoyé par le moniteur au filtre destination. Un filtre peut aussi requérir un fichier temporaire (par un message `TMPFIL`), puis, ayant obtenu son nom, y écrire (pour l'indiquer dans un message de continuation).

Chaque filtre a deux canaux d'entrées provenant du moniteur : le canal de données (contenant des messages comme figure 1) et le canal de contrôle. Ce canal de contrôle (avec un protocole simple ligne à ligne) permet au moniteur de demander à un filtre d'interrompre un traitement (par exemple il est inutile de continuer l'analyse du texte anglais si une image a suffi à rejeter la page).

Les objets embarqués dans une page HTML comme les images (balise `<img src=...`) ou les scripts (balise `<script src=...`) nécessitent qu'un filtre puisse requérir (par une commande `SUBREQ`) du moniteur un autre contenu (l'image référencée...) avant que celui-ci ne soit chargé par un navigateur client.

# 3 Les filtres spécifiques

## 3.1 Traitement du langage naturel

Pour le traitement du langage naturel (anglais, italien, espagnol, ...), le contenu est débarassé de ses balises puis envoyé (par continuation) à un filtre identificateur de langage naturel par N-grammes [17, 18]. Celui-ci détecte rapidement le langage naturel utilisé et fait continuer le traitement par un analyseur dédié.

Selon le langage, le filtrage se fait par des techniques différentes (familières aux équipes les développant).

## 3.2 Analyse des liens et des notations PICS

Une base d'URL positive et negative est prévue dans POESIA. On peut alors analyser une page HTML selon les liens (ou ancres `<a href=...`) qu'elles contient. Ces liens peuvent être statiques, mais aussi dynamiques car traités (dans un navigateur) par du Javascript. Aussi un filtre d'analyse statique de Javascript est prévu. Les hyperliens sont en effet importants dans les contenus [19, 20]

Poesia contiendra aussi un filtre pour PICS, qui transforme les notations PICS en un score de filtrage.

## 3.3 Filtrage des images

Dans POESIA nous nous intéressons au filtrage des symboles et des images pornographiques. Dans cette communication, nous présentons les premiers résultats concernant les images pornographiques.

Le filtrage des images pornographiques est un problème très ardu, en raison de la nature des images. Ces images contienent plusieurs personnes (noirs, blancs, asiatiques) dans différentes positions. Enfin, la qualité des images n'est pas toujours très bonne. La détection de la peau est une phase primordiale dans ce type de problème. Nous construisons trois modèles probabilistes pour la détection de la peau humaine dans des images. Ces modèles sont construits à l'aide d'une collection d'images dans lesquelles la peau humaine est identifiée. Nous utilisons la méthode du maximum d'entropie en fixant les lois de certaines marginales. Le premier modèle est courament utilisé. Les pixels sont indépendants entre eux. Les performances, pour la collection d'images Compaq database, mesurées par la courbe ROC, sont excellentes pour un modèle si élémentaire. Le second modèle est un markov THMM caché qui force la régularité. Les performances sont améliorées. Enfin, le troixième modèle TFOM tient compte du gradient de l'image. L'approximation des arbres de Beth nous permet d'obtenir d'obtenir une solution analytique simple pour les coefficients du modèle de maximum d'entropie. Les performances sont comparées en en utilisant la courbe ROC [21, 22].

Les expériences ont été réalisées selon le protocole suivant. La base Compaq contient 18,696 images. Cette base est divisée en deux. La première contient 2 milliards de pixels utilisée pour l'apprentissage et la seconde est utilisée pour les tests et permet de calculer la courbe ROC.

# 4 Les mécanismes de décisions

Les scores de filtrage obtenus pour chaque filtre sont enregistrés par le moniteur dans un cache de score. Ainsi, lorsqu'une page Web a déjà été accédée et filtrée, son contenu est caché par Squid et ses scores sont cachés par le moniteur ; le coût du filtrage est ainsi mutualisé.

Un mécanisme de décisions reçoit (des filtres ou du cache) des scores de filtrage pour chaque requête. Il produit une décision d'acceptation ou de rejet d'une requête, donc d'un contenu (Web, email, ...). Cette décision est alors traitée par le moniteur, qui répond à la source d'information (client ICAP pour le Web, MTA pour le mail, etc..) et interrompt les filtrages encore en cours de cette requête.

Plusieurs mécanismes de décisions sont prévus dans POESIA, notamment un mécanisme par réseau de neurone et un mécanisme Bayésien. Dans certains cas, une décision de rejet (ou d'acceptation) peut être produite avant que tous les filtres aient fini leur travail. Dans ce cas, le moniteur doit interrompre les filtres (qui traiteront d'autres requêtes). Les mécanismes de décisions sont adaptatifs et gagnent à être entrainés sur des base d'exemples. Leur configuration détermine la séverité du filtrage obtenu.

# 5 Une architecture adaptative et extensible de filtrage

Le système POESIA utilise des techniques adaptatives ou même d' apprentissage dans les filtres comme dans les mécanismes de décisions. C'est pourquoi la constitution d'une base significative d'exemples y est determinante.
L'architecture POESIA est extensible (vers d'autres domaines ou d'autres langages) par ajout de modules.
Le caractère libre du logiciel POESIA et son architecture modulaire permet l'apport d'autres contributeurs extérieurs au projet.

# Références


[1] consortium POESIA. Public opensource environment for a safer internet access. http ://www.poesia-filter.org/.

[2] Commission Européeene DG INFOSOC. safer internet action plan. http ://www.europa.eu.int/information_society/programmes/iap/index_en.htm.

[3] Stephen A. Thomas. *HTTP essentials (protocols for secure, scalable Web sites)*. Numéro isbn 0471-39823-3. Wiley, 2001.

[4] R. Fielding, T. Berners-Lee, et al.. Hypertext transfer protocol – http/1.1. Rapport technique RFC2616, W3C consortium, june 1999.

[5] Emmanuel Chailloux, Pascal Manoury, et Bruno Pagano. *Développement d'Applications avec Objective Caml*. Numéro ISBN 2-84177-121-0. O'Reilly, 2000. cf http ://www.ocaml.org.

[6] World Wide Web Consortium. Platform for internet content selection. http ://www.w3.org/PICS/.

[7] Consortium Princip. Princip project (multilingual system for detecting racist and revisionist internet documents). http ://www-poleia.lip6.fr/~princip/description.html.

[8] Jeremy Elson et Alberto Cerpa (ed.). Internet content adaptation protocol (internet draft). Rapport technique, http ://www.i-cap.org/, juin 2001.

[9] Duane Wessels. *Web Caching*. Numéro isbn 1-56592-536-X. O'Reilly, 2001.

[10] Consortium Squid. Squid cache. http ://www.squid-cache.org/.

[11] Carey Williamson. on filter effects in web caching hierarchies. *ACM Trans. Internet Technology*, 2(1) :47–77, february 2002. detailed study of multilevel [Squid] web caching.

[12] Geetha Manjunath et al.. Icap enabled squid server. http ://icap-server.sourceforge.net/.

[13] Philip Hazel. *EXIM - the mail transfer agent*. Numéro isbn 0-596-00098-7. O'Reilly, 2001. cf http ://www.exim.org.

[14] Hyokyung Bahn, Kern Koh, Sam H. Noh, et Sang Lyul Min. Efficient replacement of nonuniform objects in web caches. *IEEE Intelligent Systems*, pages 65–86, june 2002.

[15] Edith Cohen et Haim Kaplan. Refreshment policies for web content caches. *Computer Networks*, 38 :795–808, april 2002.

[16] Jun Wang, Rui Min, Yingwu Zhu, et Yiming Hu. Ucfs - a novel user-space high-performance, customized file system for web proxy servers. *IEEE Trans. on Computers*, 51(9) :1056–1073, september 2002.

[17] William B. Cavnar et John M. Trenkle. N-gram-based text categorization. Dans *Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval*, pages 161–175, Las Vegas, US, 1994.

[18] Penelope Sibun et Jeffrey C. Reynar. Language identification : Examining the issues. Dans *5th Symposium on Document Analysis and Information Retrieval*, pages 125–135, Las Vegas, Nevada, U.S.A., 1996.

[19] Gary William Flake, Steve Laurence, C. Lee Gies, et Frans M. Coetzee. Self-organisation and identification of web communities. *IEEE Computer*, pages 66–71, march 2002.

[20] Amanda Spink, Bernard J. Jansen, Diertmar Wolfrwm, et Tefko Saracevic. From e-sex to e-commerce : Web search changes. *IEEE Computer*, pages 107–109, march 2002. web search are less for sex, more for commerce, people or non-english (evolution of web search queries from 1997 to 2001).

[21] B.Jedynak, H.C.Zheng, M.Daoudi, et D.Barret. Maximum entropy models for skin detection. Dans *Indian Conference on Computer Vision, Graphics and Image Processing*, december 2002. (accepted paper).

[22] B.Jedynak, H.C.Zheng, M.Daoudi, et D.Barret. Maximum entropy models for skin detection. Rapport technique XIII (vol.57), PUB.IRMA, 2002.