École Polytechnique de Montréal, Québec, Canada

Rapport du Projet de Maîtrise

Portage d'un navigateur web sur AmigaOS 3.x

Aymen Zalila
zalila.aymen@polymtl.ca

Travail dirigé par : Professeur. Yann-Gaël Guéhéneuc
Directeur d'études : Steven Chamberland

Hiver 2016

Dédicaces

Je dédie ce travail à :

Mes grands-parents, paix à leurs âmes

Vous êtes toujours dans mes pensées

Mes parents bien-aimés

Mes deux très chères sœurs

Mes nièces adorées

Et toutes les personnes que j'ai rencontrées à Montréal

Et qui ont impacté ma vie durant ces 2 dernières années.

Aymen Zalila

Remerciements

Je remercie Le professeur Yann-Gaël Guéhéneuc pour l'opportunité qu'il m'a donner pour réaliser ce projet, de m'avoir initier au monde Amiga, et de m'avoir fait rencontrer de gens extraordinaires en fesant partie de son quipe.

Je remercie l'École Polytechnique de Montréal, pour la formation de très haute qualité dont j'ai bénéficié durant ma maîtrise.

Je remercie les membres des équipes SoccerLab et de PtiDej, d'avoir étaient des collègues professionnels, avec qui je me suis noué d'amitié.

Je remercie la communauté Amiga d'avoir répondu à mes questions et je salue son engagement exemplaire envers sa passion.

Je remercie, enfin, ma famille et mes amis d'être toujours présents à mes côtés et de m'avoir soutenu dans les moments difficiles.

Merci

Table des matières

Table des figures

Liste des tableaux

Introduction

Dans le cadre de mes études supérieures à l'École Polytechnique de Montréal, je suis amené à réaliser un projet de maîtrise pour finaliser mon parcours. Ce projet consiste à étudier la faisabilité et d'essayer de porter un moteur de rendu HTML efficace sur la plateforme AmigaOS3.x.

AmigaOS est un système d'exploitation qui existe depuis les années 1980. C'est un système simple, efficace et proche des systèmes d'exploitation temps rel. Il est activement entretenu et appartient à un créneau du système d'exploitation pour ordinateurs de faible puissance. Actuellement, la principale barrière à l'entrée est le manque d'un navigateur Web avec le support de fonctionnalités modernes : HTML4, CSS et JavaScript.

L'objectif du projet est d'évaluer les navigateurs Web à code source libre existants, pour étudier leur portabilité, et pouvoir concevoir et mettre en œuvre les changements nécessaires pour un navigateur Web pour AmigaOS.

Dans ce document, on commence par parcourir l'histoire de Amiga où l'on verra son impact sur l'évolution de la technologie. On définit, en second, la compilation croisée, une technique incontournable pour réaliser les objectifs, ainsi que les technologies nécessaires. On parcourra ensuite les navigateurs Web existants et les technologies qui leur sont affilées. Enfin, on discutera des possibilités qui sont offertes et des choix faits durant cette étude.

Ce projet présent un recensement des navigateurs Web à code source libre qui existants et une description des technologies qui les animent. Il offre un tutoriel détaillé de la préparation d'un environnement de compilation croisée à destination de AmigaOS3 68k et ouvre la porte à des tests futurs sur les différents navigateurs. Enfin, il apporte une contribution aux utilisateurs de Amiga avec un environnement de compilation prêt à l'emploi et la plus récente version du navigateur NetSurf dédiée à AmigaOS 3.x.

Chapitre 1

L'histoire de Amiga

Dans ce premier chapitre, on passe en revue l'historique des technologies Amiga. Depuis sa propulsion sur les devants de la scène à ses multiples difficultés financières, l'importante communauté de passionnés qui la supporte depuis plus de trente ans et l'état actuel de son héritage [4].

1.1 Les débuts

L'histoire de Amiga commence bien avant sa création. Durant les années 1970, Jay Glenn Miner, un ingénieur en électrotechnique, participe à la création de consoles de jeux 8 bits chez Atari. Ces consoles avaient une architecture de base qui allégeait la tâche du processeur en utilisant des circuits spécialisés pour gérer le son et le graphique. Le succès de ces consoles satisfaisait amplement la société, ce qui n'était pas de cas de Miner. Ce dernier proposa de développer un ordinateur basé sur les processeurs 68000 (ou 68k) de Motorola, encore chers à l'époque. Cette divergence de points de vue et le refus de verser le bonus prévu poussèrent Jay Miner, Larry Kaplan (programmeur en chef) et d'autres ingénieurs à quitter Atari [14].
Quelques années plus tard, Kaplan reprit contact avec Miner pour lui proposer de développer une console qui bousculerait le marché des jeux vidéo. Bert Braddock, patron de Miner dans la société Zymos et David Morse premier actionnaire du projet ont rejoint l'équipe.
En septembre 1982, une entreprise nommée initialement "Hi-Toro", puis "Amiga Inc.", voit le

FIGURE 1.1 – Premier logo Amiga

jour à Santa Clara, en Californie. L'objectif de ses investisseurs était donc de créer une console de jeux vidéo pour concurrencer Atari et Nintendo. Jay Miner pour sa part, garda son rêve de créer une machine avec des ports d'extension et utilisant les microprocesseurs 68k qui révolutionnerait le marché des micro-ordinateurs [15].

La société se lance dans la conception et le développement de manettes et de joysticks ainsi que des jeux vidéo pour Atari dans le but de se faire connaître par le marché. En parallèle, une équipe travaillait sur une machine et un système d'exploitation propriétaires, baptisée Lorraine. Au cours de l'année 1983, le marché des consoles de jeux commençait à s'effondrer et à son opposé celui des ordinateurs émergeait. Amiga Inc. délaissa alors ses développements pour Atari et se concentra sur son projet d'ordinateur. Les équipes de développement conçurent plusieurs prototypes de circuits intégrés : Agnus , Daphne et Portia ainsi que Labdec pour brancher les cartes ensemble dans le but de les présentés au Consumer Electronics Show. Des soucis financiers poussèrent Amiga à signer un accord avec Atari, permettant à cette dernière d'accéder aux puces de la Lorraine [16].

1.2 L'arrivée de Commodore

En janvier 1984, l'équipe réalise une première présentation de son prototype "Lorraine/Amiga PC" avec l'objectif d'attirer de nouveaux investisseurs. En août 1984, Amiga Inc. fût racheté par Commodore, qui crut en son potentiel et y investit en faveur de ses propres projets, dans le but d'une commercialisation grande publique future et pour rattraper les sorties de l'Atari 520 ST et du Macintosh de Apple [17].

Le 23 juillet 1985, le Amiga 1000, est présenté dans une grande cérémonie à New York avec une démonstration de Andy Warhol. Le Amiga 1000 est considéré comme une technologie en

avance sur la concurrence. Il était équipé du microprocesseur Motorola 68000 auquel étaient ajoutés des processeurs spécifiques pour le son, le graphisme et l'animation. Son système d'exploitation, le Workbench, présentait une interface graphique en couleurs (4 096 couleurs), un système de son avancé, il était multitâches et la machine offrait plusieurs équipements et possibilités d'extensions inédites [18].

L'Amiga était attractif, mais connaissait des débuts difficiles principalement du fait de son prix élevé, son marché limité à l'Amérique du Nord et qu'il n'était compatible avec rien et qu'il fallait créer toute une logithèque de zéro.

Dans les années qui suivent, Commodore-Amiga développa de nouvelles machines. Les modèles Amiga 2000 (haut de gamme) et Amiga 500 (entrée de gamme) sont sortis en 1987, avec une version mise à jour du Workbench et fût enfin vendu en Europe. Amiga se fit une place dans le marché mondial, et plusieurs nouveaux périphériques matériels, développés entre autres par de compagnies tiers, fient leur apparition et vinrent apporter leur contribution à l'extensibilité de ces machines.

1.3 l'âge d'or

Les ventes des Amigas commenceraient à décoller, plusieurs jeux vidéo de qualité furent développés sur ces plateformes.Des titres comme "Defender Of The Crown", ce jeu remporta la rcompense du meilleur graphisme de la Software Publishers Association de 1986, ou "Hybris" et "Battle Chess" qui était irréalisable sur les machines de la concurrence. Amiga est devenu la référence dans le domaine du graphisme et de l'audio. Les professionnelles de la musique et du cinéma (avec VideoToaster, LightWave 3D et Babylon 5) ont en fait leur outil de travail et la communauté de fans commençait à apparaître. Commodore International tenait enfin sa mine d'or.

En 1988-89, l'expansion continue avec la sortie de plusieurs jeux et logiciels, la mise à jour du Workbench, et l'intégration les nouveaux modèles de microprocesseur de Motorola. Commodore recommence à lancer ses propres ordinateurs. Mais la direction décida de diminuer le budget alloué à la recherche pour booster ses profits [19].

Pour la nouvelle décennie, Amiga produit la deuxième génération de machines et de système

d'exploitation. Des améliorations d'anciennes machines ainsi que le Amiga 3000 et le CDTV (Commodore Dynamic Total Vision), sont introduits dans le marché et plusieurs logiciels révolutionnaires à l'instar de VoRecOne (premier logiciel de reconnaissance vocale), TechnoSound Turbo et Action Replay, ont été développés. Mais l'année 1990 vit l'ascension fulgurante de Windows avec sa version 3.0 et sa propagation dans les foyers américains. Ainsi que l'apparition des consoles 16bits de Sega et Nintendo. Amiga était bousculé sur tous les front.
Une troisième génération de Amigas équipée de nouveaux processeurs et tournant sous AmigaOS 3.0, un tout nouveau système d'exploitation, a été introduite en 1992. Cette génération était populaire par sa puissance, sa grande variété de logithèques et une multitude d'extensions possibles.
L'année suivante, le Workbench 3.1 était lancé (dernier OS publié par Commodore) et la console Amiga CD32 commençait sa production, la première console 32bits avec support CD. Mais cette machine apporta son lot de problèmes, avec notamment un conflit juridique qui empêcha sa commercialisation aux États-Unis. Commodore commença à entrer dans une période d'instabilité financière et les salaires élevés des dirigeants et les investissements exorbitants dans des sponsors sportifs ont précipité la chute.

1.4 La fin de Commodore

En 1994, les différentes filiales et usines de Commodore international à travers le monde commençaient à fermer leurs portes à cause des réclamations de paiement des différentes banques. Commodore passa en liquidation volontaire. Malgré un avenir incertain, les produits Amiga continuaient à évoluer avec le Amiga 4000, de nouveaux supports pour le CD32, un nouvel AmigaOS 3.1 et de nouvelles productions des éditeurs de logiciels. La société allemande Escom a initié des négociations pour le rachat de Amiga, mais huit mois après la liquidation de Commodore aucune avancée n'a été faite dans le dossier et les espoirs des utilisateurs dans la sortie d'un nouvel Amiga 5000 et un Workbench 4.0 commençait à s'envoler [20].
Un affrontement pour le rachat de Commodore, Amiga et leurs droits éclata entre plusieurs entreprises, principalement Dell et Escom. Cette dernière remporta la bataille le 21 avril 1995.

Une nouvelle ère est lancée.

1.5 Une Nouvelle Ère

Figure 1.2 – Logo Amiga 1995

Escom veut relancer Amiga. De nouveaux modèles équipés des processeurs 68060 sont prévus, de nouveaux magazines à l'effigie de la marque, qui avait un nouveau logo, sont publiés. Mais la communauté réclamait le Amiga à base de l'architecture Power PC et un AmigaOS 4. Les Amigas commençaient à se faire vieux devant les PC dotés de Pentium et les Mac de PowerPC 601.

Malgré un espoir de renouveau et de retour au premier plan, les difficultés financières que rencontre Escom après le lourd investissement (le rachat et la recherche et développement,) et l'image ternie de Amiga après la faillite de Commodore, fient annuler le contrat de projet PowerUP. Phase 5 la société qui était en charge de ce dernier, continua tout de même le développement et est devenue par la même occasion le plus grand joueur du monde Amiga.

Plusieurs autres sociétés, comme PIOS et ProDAD, se sont lancées dans le développement de clones de Amiga et de ses systèmes d'exploitation. Ainsi Amiga est devenu une technologie entretenue et développée par des acteurs tiers et la communauté, sans avoir de véritable représentant ou propriétaire.

À partir de 1996, la communauté commence à se diviser, les avis à propos des changements à apporter aux supports et aux systèmes sont partagés, les projets se multiplient et les visions s'entrechoquent. Plusieurs projets visant à permettre à Amiga de se connecter à l'Internet sont nés [21]. Les brevets, les licences et les droits d'exploitation de Amiga sont rachetés à plusieurs reprises par différentes sociétés à travers le monde (VIScorp, Gateway 2000, Hyperion Entertainment).

1.6 Place à la Communauté

Durant les années Commodore, Amiga s'est construit l'image d'une technologie évolutive, en avance sur son temps et s'est fait aimer par une vaste communauté. Cette communauté reprit le flambeau après les faillites des sociétés qui se sont succédé à la tête de Amiga. Et plusieurs projets prometteurs ont vu le jour et sont entretenus jusqu'à maintenant par des passionnés, dans le but de maintenir l'héritage et de mettre le Amiga au niveau des technologies les plus récentes [13] [22].
En l'absence d'un dirigeant sérieux, plusieurs chemins se sont tracés. Parfois entrepris par de société comme le duo Phase 5/Haage & Partner, par des professionnels qui ont collaboré avec Commodore et Amiga ou des fans.
Les projets les plus importants ont été :

Les versions 3.5 et 3.9 du Workbench, sorties respectivement en 1999 et 2000, ont été dévelop-pées par la société allemande Haage & Partner. Ces systèmes d'exploitation tournaient sous des processeurs Motorola 68020.

AmigaOS 4 est un projet visant à porter le système d'exploitation sur des supports PowerPC. Ce projet est initié par Amiga Inc. et le développement est laissé aux soins de Hyperion Entertainment sous licence en 2001. La première version de cet OS paru fin 2006 était quasi-entièrement PowerPC, mais permettait d'émuler l'exécution de code 68k et donnait la liberté aux développeurs d'utiliser l'ancienne logithèque.

AmigaOne est aussi un projet mené par Hyperion Entertainment. Il vise à créer une machine Amiga munie de PowerPC et pouvant supporter le Workbench version 4. Après de multiples problèmes, il a vu finalement le jour en 2003.

MorphOS est un projet de système d'exploitation pour PowerPC créé en 1999 et est présenté comme une évolution de AmigaOS Classic. Il est développé par The MorphOS Development Team. Il est actuellement le système d'exploitation Amiga sur PowerPC le plus portable qui existe.

Pegasos est encore un projet de carte mère utilisant un processeur PowerPC. Elle est destinée au système MorphOS et autres. Elle a été développée par la société bplan. La deuxième génération, le Pegasos II sont entrés en production en 2006.

AROS Amiga Research Operating System est un OS à code source libre, développé dans le but de rendre AmigaOS 3.1 portable et compatible avec les machines dotées de processeurs x86. Le projet a commencé en 1995 et a donné naissance à plusieurs distributions : AspireOS, Broadway et la plus avancée Icaros Desktop.

Amithlon visait à créer un émulateur des Amiga Classics pour les x86. Il était facile à installer et exceptionnellement rapide, mais le projet fut arrêter pour des problème légaux.

UAE est aussi un émulateur pour Amiga. Sa première version est sortie en 1996 et était le premier émulateur à faire tourner AmigaOS sur les portables modernes. WinUAE support les PowerPC depuis 2014.

ACube System, née du regroupement de 3 sociétés italiennes, a développé les SAM440. Des cartes visant à supporter AmigaOS 4 et remplacer le AmigaOne. Ces cartes ont été développées un certain temps avec un franc succès, mais leurs capacités ne répondaient pas à l'usage des développeurs réguliers. [37]

La communauté Amiga ne se compose pas uniquement de fans ou de groupes isolés. Plusieurs professionnels et entreprises se sont spécialisés et ont pris pour mission de raviver la flamme de l'univers Amiga malgré un espoir de plus en plus mince.

1.7 Actuellement

Les 30 ans de Amiga (juillet 2015) furent l'occasion de rassembler les différentes communautés de par le monde. Plusieurs événements ont été organisés pour cette date marquante de l'histoire de cette technologie. Les plus importants étant celui de Mountain View (Californie) organisé au Computer History Museum [2] et en Europe celui d'Amsterdam [3].
Malgré que Amiga ne soit plus aussi populaire, et que connaître son existence releve de l'exploit pour la nouvelle génération, la communauté enregistre encore régulièrement de nouveaux adeptes.

Depuis octobre 2009, Hyperion Entertainment détient officiellement tous les droits de AmigaOS. Elle poursuit encore le développement de ses systèmes. L'AmigaOS 4.1 Final Edition est sortie au début de l'année 2015. Et OS 5 est en cours de développement.
Les émulateurs UAE sont constamment mis à jour et maintiennent l'utilisation des Amiga Classics, ainsi que la dernière génération, en activité.
Le plus gros problème que rencontre la communauté est le manque de matériel. Les équipements qui peuvent faire tourner des Amigas ne sont quasiment plus produits et les anciennes machines sont devenues rares qu'il est litéralement impossible de s'en procurer, sauf si l'on est prêt à payer le prix fort.
Le second plus gros problème est le manque de navigateurs Web sur les plateformes classiques de Amiga. Les quelques navigateurs qui existent encore sont vieux et ne supportent pas les sandards comme HTML4 et CSS 2 et ceux en développement rencontrent des problème d'intégration. Plusieurs applications sont développées pour faire entrer les Amigas dans la nouvelle ère de connectivité. Par exemple, des applications de synchronisation avec les équipements mobiles ou comme AmiCloud qui vise à introduire le Cloud.

Ce survole de l'histoire de Amiga et de son héritage, est incontournable pour introduire cette étude. Durant ces 4 mois, en parlant de Amiga avec collègues et autres connaissances, on s'est rendu compte que la nouvelle génération, y compris moi, n'a aucune conscience de l'existence de cette technologie et du rôle important qu'elle a joué dans l'évolution du monde informatique et multimédia, avec sa vision de gestion des ressources, son architecture extensible et sa position à la pointe de la technologie.
Dans le chapitre suivant, on présente la compilation croisée et les technologies que l'on utilise pour réaliser ce projet.

Chapitre 2

Environnement et techniques de travail

Ce second chapitre introduit la compilation croisée, ainsi que les différentes techniques et technologies choisies à cet effet. Ce concept constitue une étape cruciale et incontournable dans le travail que nous voulons réaliser.

2.1 La compilation croisée

2.1.1 Concept et définition

La compilation croisée (ou *cross-compilation* en anglais) est une technique qui permet à l'aide d'une machine avec une architecture spécifique et un système d'exploitation donné de compiler un programme, et donc créer un exécutable, pour une autre architecture ou un autre système d'exploitation [7]. Dans notre cas, nous utilisons une machine Windows ou Linux 64 bits pour compiler un programme destiné aux systèmes AmigaOS 3.x en 68k.
La compilation croisée peut être motivée par plusieurs raisons dont : le gain de temps, elle permet d'éviter de faire des redémarrages répétés lorsque l'on dispose d'outils de compilation et de développement sur une machine et non sur la machine de destination, ou pour profiter de la puissance de calcul de la première machine. Il se peut aussi que l'on ne dispose pas des licences adéquates de la machine ou le système de destination, cela permet donc de contourner ce type de contraintes.

La compilation croisée permet en fait de créer un compilateur spécifique. Il est capable de jouer le rôle d'interprète entre deux architectures ou systèmes différents.
La partie qui suit contient les logiciels, émulateurs et bibliothèques utilisés lors du projet, afin de créer le compilateur voulu et de pouvoir tester les exécutables.

2.2 Environnement de compilation

Pour l'environnement de compilation, on a besoin d'un compilateur et d'un interpréteur de ligne de commande Shell pour exécuter les scripts nécessaires.

2.2.1 GCC

GNU Compiler Collection est le compilateur utilisé durant ce projet. Les avantages à utiliser GCC sont qu'il est compatible avec la plupart des bibliothèques et outils à code source libre du marché. Il est généralement utilisé dans le développement à code source libre. Il offre la majorité des compilateurs les plus utilisés actuellement.

2.2.2 Windows

Le DOS de Windows à l'état brût ne permet pas de faire de la compilation croisée. Pour cela, l'utilisation d'outils spécifiques et nécessaires. Plusieurs choix sont offerts, mais l'utilisation de l'émulateur Cygwin semble le plus adéquate, car plus complet et plus flexible.
Il faut savoir qu'il existe plusieurs IDEs créées spécifiquement pour ce genre de compilation. On peut nommer AmiDevApp [1], un environnement de développement dédié à la compilation croisée sur Windows et vers les systèmes Amiga. Il n'est tout de même pas conseillé d'utiliser ce type d'environnement pour compiler des programmes de grande taille. Ils sont plus adaptés au développement de nouveaux programmes.

2.2.3 Cygwin

La collection Cygwin de Cygnus Solutions permet de profiter des avantages d'un environnement système UNIX sur Windows.
Il est fortement recommandé d'utiliser Cygwin dans sa version 32 bits même sur une machine 64 bits. La version 32 bits est plus compléte et contient tous les paquets nécessaires pour émuler parfaitement un environnement UNIX.
Pour répondre aux besoins de la compilation prévus, les paquets suivants doivent être téléchargés à l'installation ou à la mise à niveau de Cygwin :

- le paquet Devel est le plus important. Plus précisemment, on a besoin des bibliothèques suivantes :
  - tous les paquets contenant "gcc"
  - ceux liés à binutils
  - flex 2.5.39-1
  - bison 3.0.4-1
  - make 4.1-1
  - epoch2 : gcctools-epoch2-autoconf 2.64-2 et gcctools-epoch2-automake 1.11.6-1
  - et autoconf 13-1
- le paquet Editor, surtout les bibliothèques liées à make,
- le paquet Doc est aussi vivement conseillé.

2.2.4 Linux

Pour pouvoir compiler sous Linux, il est nécessaire d'installer quelques paquets. La commande suivante suffit à préparer l'environnement.

```
$ sudo apt-get install apache2 g++ python subversion gperf \\
  make devscripts fakeroot flex bison autoconf autoconf2.64
```

2.3 Environnement de test

Pour pouvoir tester les programmes compilés, on a besoin d'un système d'exploitation Amiga et d'un émulateur pour pouvoir le faire marcher. Les outils utilisés, lors du projet, sont la bibliothèque AmigaForever dans sa version Plus de 2013 et l'émulateur AmiKit 8.
AmigaForever est la compilation officielles des émulateurs Amiga. Elle a été développée par Cloanto. Elle contient les images de tous les Workbenchs officiels (+ Kickstarts) ainsi que ceux de plusieurs jeux vidéo qui on fait la gloire de Amiga dans les années 80.
Elle inclue un lecteur qui utilise la puissance de WinUAE pour l'émulation, tout en affichant

FIGURE 2.1 – Amiga Forever

ça propre interface. Elle est disponible en trois éditions différentes, qui offrent plus ou moins de contenu et d'options selon le besoin.

2.3.1 AmiKit

AmiKit est un environnement complet, réserver au monde Amiga. Il offre toute une panoplie de programmes pré-installés et permet une grande liberté de configuration de l'émulateur et du système émulé.
Il offre une interface graphique avancée, intuitive et agréable à utiliser. Il offre donc une expe-

rience complete et totalement inédite aux utilisateurs d'Amiga. Il est disponible sur Windows, Mac OS, Linux et Android.

FIGURE 2.2 – AmiKit 8

Chapitre 3

Les navigateurs Web et technologies utilisées

On passe en vue, dans cette partie, les différents navigateurs Web du marché ainsi que les technologies (moteurs de rendu HTML et moteurs ECMAScript) choisies par leurs développeurs pour répondre aux besoins des utilisateurs et aux standards du Web. Bien que l'on recherche une technologie open source, ce recensement contient quelques logiciels propriétaires qui peuvent guider les choix.
Depuis l'arrivée de l'Internet, les développeurs et utilisateurs de Amiga ont essayé de faire surfer leurs machines sur cette vague. Dans les années 90, NCSA Mosaic avait même une version dédiée aux Amigas, nommée AMosaic.

3.1 Les navigateurs Web

Les navigateurs Web sont des logiciels extrêmement repandus actuellement, vu la place qu'occupe internet dans le quotidien. Il en existe de toutes sortes, pour tout matériel (ordinateurs, tablettes, téléphone…) et pour les différents systèmes d'exploitation (Windows, Unix, Mac OS…). Les navigateurs Web disposent de composants leur permettant de communiquer avec le réseau, d'un moteur de rendu pour les standards du Web, d'un moteur ECMAScript pour pouvoir exécuter les scripts des pages et d'une interface utilisateur.

Les navigateurs les plus utilisés actuellement sont Google Chrome, Safari, Internet Explorer, Firefox et UCBrowser (ce dernier est surtout présent en Chine et au Japon).

Safari est le navigateur Web développé par Apple Inc., sa première version est parue le 7 janvier 2003. Il a été développé dans le but de rompre la dépendance envers Internet Explorer de Microsoft. Il doit principalement son importante part de marché à la popularité des produits d'Apple. Il reste souvent prisonnier de ces derniers et est rarement utilisé sous d'autres systèmes.
Safari est particulièrement agréable à utiliser sur les OS d'Apple. Il s'intègre parfaitement et le plus rapide sur ces systèmes. Il intègre toutes les fonctionnalités des navigateurs modernes comme les onglets, les favoris et les extensions. Il bénéficie aussi d'un gestionnaire de flux RSS très poussé.

Google Chrome est un navigateur développé par Google. Il est basé sur le projet à code source libree Chromium (sous licence BSD). Il est actuellement le navigateur le plus utilisé dans le monde.
Dès ses débuts en 2008, Chrome a révolutionné le monde des navigateurs en popularisant les systèmes d'onglets. Il est particulièrement rapide et adapté à tous les systèmes d'exploitation du marché.

Firefox est le navigateur Web open source le plus populaire depuis qu'il a pris la relève de son prédécesseur Netscape. Il est développé et maintenu essentiellement par The Mozilla Foundation, aidée de nombreux bénévoles. Sa première version est apparue en septembre 2002.
Firefox est connu pour sa conformité au règle et standard imposés par la W3C. Il est très apprécié par ses utilisateurs. Il se déclare sécuritaire et fervent protecteur des données privés.

Table 3.1 – Comparatif des navigateurs 1/2

	Plateformes	Langage	Plugins/extensions	Licence
Safari	Mac, Windows (jusqu'en mai 2012) et iOS	C++ et Objective-C	supportés	propriétaire
Chrome	Windows, Mac OS, GNU/Linux, Android et iOS	C++, Java, JavaScript et Python	supportés	propriétaire
Firefox	Linux, Windows, OS X, Firefox OS, Ubuntu Touch et Android	C++, JavaScript4 et XBL	supportés	MPL/GPL/LGPL
NetSurf	RISC OS et POSIX	C	non supportés	GPL
IE / Edge	Windows / Windows10	C++	supportés (Edge)	propriétaire
Elinks	Linux, Microsoft Windows, Mac OS, UNIX, RISC OS, BeOS et Berkeley Software Distribution		non supportés	GPL v2.0
OWB	AmigaOS, AROS, MorphOS	C++	non	BSD
iBrowse	Amiga	C	non	propriétaire
Dillo	plates-formes UNIX	C++	non supportés	GNU GPL v3

NetSurf est un navigateur libre développé par The NetSurf Developers sous la licence GNU GPL principalement pour les plates-formes RISC OS. À l'origine, NetSurf-Browser était écrit pour tourner sur des télévisions, les PDA et les téléphones portables. Sa conception est destinée à être compacte et peu gourmande en mémoire.

NetSurf est développé par une communauté de programmeurs issants des milieux différents et très actifs. Il est écrit en C et est axé sur la portabilité. Il est considéré comme pionnier du concept de page Web *thumbnailing*, qui permet de créer des copies miniatures des images et vidéos pour alléger la page.

Internet Explorer et Microsoft Edge sont deux navigateurs développés par Microsoft. IE était le plus important du marché jusqu'à 2012. Il a longtemps influencé le marché et la manière de développer pour le Web jusqu'à la rébellion des autres acteurs du marché du Net. Ses concurrents ont alors dénoncé l'impact néfaste que ce genre de pratique avait sur l'évolution du domaine et ont donc poussé la W3C à parfaire ses standards et ses réglementations. Microsoft Edge est devenu le nouveau navigateur propriétaire de Microsoft à partir de 2015. Il est perçu comme un signe de renouveau par Microsoft et veut donner une nouvelle image à la navigation sur Windows. Il intègre toutes les fonctionnalités attendues d'un navigateur moderne et répond à la plupart des standards actuels.

TABLE 3.2 – Comparatif des navigateurs 2/2

	Site officiel	dernière version	HTML5/CSS3	Part de marché
Safari	`apple.fr/safari`	2 juin 2014	oui	18,6% (janvier 2016)
Chrome	`google.com/chrome`	21 janvier 2016	oui	44,6% janvier 2016
Firefox	`mozilla.org/firefox`	11 février 2016	oui	9,5% janvier 2016
NetSurf	`netsurf-browser.org`	15 mars 2015	HTML4 / CSS2.1	
IE / Edge	`microsoft.com/ie` et `browserfordoing.com`	9 décembre 2015 (1ER) et 12 novembre 2015 (Edge)		12,9% janvier 2016
Elinks	`elinks.or.cz`	30 octobre 2012		˜0%
OWB	`owb.mikendezign.com`	15 April 2014	HTML5 / CSS3	
iBrowse	`ibrowse-dev.net`	22 décembre 2006	HTML4	majoritaire pour Amiga jusqu'à l'arrêt de développement
Dillo	`dillo.org`	3.0.5 30 juin 2015	HTML4/non	˜0,01%

Links et Elinks sont des navigateurs textuels. C'est-à-dire que leur affichage est uniquement composé de texte. Le second est basé sur premier.

Elinks a poussé le concept plus loin en affichant les pages Web comme sur un navigateur

standard en utilisant seulement les caractères textuels et les couleurs de l'arrière-plan. Mais le développement de ce dernier est suspendu depuis 2012. Links par contre est régulièrement mise à jour par ses développeurs.

Origyn/Odyssey Web Browser (OWB) était appelé à ses débuts Origyn Web Browser et développé par Sand-labs. Il a été renommé en OdysseyWB par le développeur Fabien Coeurjoly après son abandon par son équipe de développement. C'est un navigateur qui utilise le composant WebKit de Apple comme moteur de rendu HTML et moteur JavaScript.
OWB tire sa popularité des systèmes d'exploitation Amiga et AmigaOS-like, où il est considéré comme étant le navigateur Web le plus mature. Il support HTML5 et Flash. Il a aussi été porté sur d'autres systèmes plus populaires, ainsi que des systèmes embarqués.

iBrowse est un navigateur Web conçu pour les ordinateurs Amiga. Il est réécriture de AMosaic et a été développé à l'origine par la société Omnipresence. Son développement a survécu à la société grâce à Stefan Burström. iBrowse interprète plusieurs standards, dont HTML4 et JavaScript.
Le développement de iBrowse est intérempu depuis 2006 et il n'est plus vendu depuis 2007. Des patches ont quand même été publiées en 2008 et 2011. Le développeur a même annoncé en 2014 et en 2016 une version 2.4, mais la communauté attend toujours.

Dillo est un navigateur Web minimaliste particulièrement adapté pour les vieux ordinateurs, les petites configurations ou les systèmes embarqués. Développé par Jorge Arellano Cid, un développeur chilien, à la fin de l'année 1999 [10].
Les objectifs principaux du projet Dillo sont la sécurité et la vie privée, ainsi que la performance accrue.

3.2 Les moteurs de rendu

Un moteur de rendu est en fait un composant logiciel utilisé par les navigateurs Web. Il contient un ensemble de bibliothèques qui sont utilisées pour afficher à l'écran le contenu des pages Web. Ils sont propres à chaque navigateur [31], même si certains reprennent ceux développés pour d'autres ou qu'il existe des moteurs de rendus génériques. Leurs développements sont étroitement liés à ceux de leurs conteneurs.

WebKit (WebCore) est un embranchement de KHTML développé par Apple et Nokia. Il intègre les bibliothèques WebCore et JavaScriptCore. Il est le composant de navigateur le plus complet du marché. Il fait face dernièrement aux mêmes accusations qu'a rencontré le navigateur IE de Microsoft en 2006. Sa popularité pousse les développeurs Web à ne considérer que lui et à introduire ses balises dans leur propre code, ce qui pousse la concurrence à être obligée de le suivre.
WebCore est le moteur de rendu HTML de WebKit.

Blink est un embranchement de WebKit développé par l'équipe responsable du projet Chromium au sein de Google et Opera Software depuis 2013. Ils ont divergé du projet WebKit suite à des désaccords avec Apple et sa politique de conduite du projet. Google déclare que Blink est une version améliorée de WebKit, avec un code plus net et clair.

Gecko lancé par la fondation Mozilla en 1998 pour Netscape, respecte les standards du Web et les recommandations du W3C. Il est compatible avec toutes les plateformes du marché et est présent dans tous les produits de Mozilla.

NetSurf (engine) est le moteur de rendu HTML de NetSurf-Browser. Il fait partie intégrante de NetSurf et a été développé spécifiquement pour lui de zéro. Il est donc aussi activement

maintenu par la communauté.

TABLE 3.3 – Comparatif des moteurs de rendu 1/2

	Intégré à	Langage	Licence	Libre
WebKit/WebCore	Safari, tous les intégrés dans tournant sur iOS, OWB, JavaFX…	C++	BSD et GNU LGPL	oui
Blink	Chrome, Chromium, Opera, Vivaldi	C++	BSD et GNU LGPL	oui
Gecko	Firefox, Thunderbird…	C++	MPL, GPL et LGPL	oui
NetSurf (engine)	NetSurf	C	GPL	oui
Trident et EdgeHTML	Microsoft Edge	C++	propriétaire	non
Servo	rien (expérimental)	Rust	MPL	oui
ELusive	Elinks		GPL v2	oui
Dillo (engine)	Dillo	C / C++	GPL v3	oui

Trident et EdgeHTML sont les deux moteurs de rendu développés par Microsoft pour ses navigateurs IE et Microsoft Edge, respectivement.
Trident a été abandonné en faveur de EdgeHTML. Il était critiqué par les utilisateurs pour son manque de performances. Il atteint le score de 100% au test Acid3 en retard par rapport à la concurrence.
EdgeHTML est le moteur de rendu développé par Microsoft pour son navigateur Microsoft Edge, c'est une embranchement de Trident. Il répond parfaitement aux exigences des standards actuels.

Servo est un moteur de rendu expérimental pour navigateur Web en cours de développement par Mozilla et Samsung. Il n'est pas prévu pour remplacer Gecko.
Le prototype vise à créer un environnement optimisant l'efficacité énergétique tout en maximisant le parallélisme, dans lequel les composants (tels que le rendu, le parsing HTML, le

décodage des imagesfi) sont gérés dans des tâches isolées.

ELusive est un moteur intéressant vu sa capacité à convertir les éléments graphiques d'un page Web en caractères textuelle et de les affichés d'une manière assez fidèle au rendu original. Il prend en charge partiellement du JavaScript. Mais étant donné que le projet Elinks est en arrêt de développement, son moteur de rendu l'est aussi.

TABLE 3.4 – Comparatif des moteurs de rendu 2/2

	Site officiel	Dernière version	HTML5/CSS3	Test Acid3
WebKit/WebCore	`Webkit.org`	v601.4.4 15 janvier 2016	oui	100% le 27 mars 2008 (sur Opera)
Blink	`chromium.org/blink`		oui	100%
Gecko	`developer.mozilla.org/en/docs/Gecko`		oui	100%
NetSurf (engine)	`netsurf-browser.org`		HTML4 / CSS2.1	ne passe pas le test
Trident et EdgeHTML		v13.10586 12 novembre 2015	oui	100% le 20 septembre 2011 (date de mise à jour du test)
Servo	`github.com/mozilla/servo` ou `servo.org`			22% puis erreur

3.3 Les moteurs ECMAScript

Un moteur ECMAScript est un programme qui exécute un code source écrit dans une version du standard de langage ECMAScript (ex : JavaScript).
Le JavaScript est le script le plus représentatif des scripts ECMA. Il a été révélé par le navigateur Netscape et sandardisé par la suite. Des dérivés comme JScript existent mais ne sont pas à considérer vu leur faible utilisation.

JavaScriptCore (SquirrelFish) est basé KJS, le maoteur ECMAScript du projet KDE. Il est moteur JavaScript utilisé par WebKit.

V8 développé par Google, est à code source libre et fonctionne pour les architectures x86 et ARM

Chakra développé par Microsoft pour IE. Il est considéré comme l'un des moteurs ECMAScript les plus rapides sur le marché. Microsoft a publié une version à code source libre en janvier 2016 appelée ChakraCore.

SpiderMonkey écrit par Brendan Eich à Netscape Communications et délivré plus tard comme un logiciel open source. Il est maintenant maintenu par la Fondation Mozilla.

Duktape est un moteur JavaScript facilement intégrable, mettant l'accent sur la portabilité et son empreinte compacte. Il est léger et facile à intégrer dans un projet en C/C++.

V7 Est une partie de la plateforme Smart.js et est considéré comme étant le plus léger des moteurs JavaScript au monde.

Ce chapitre a présenté les différentes technologies qui animent le monde des navigateurs Web du moment.

Dans le chapitre suivant, il sera question de choisir quelle technologie adoptée pour aMIGAos 3.x.

Table 3.5 – Comparatif des moteurs de JavaScript

	Intégré à	Langage	Licence	Libre	Site officiel	ECMAScript version	Dernière version
JavaScriptCore	`Webkit.org`	C++	GNU LGPL	oui		E5.1	
V8	Chromium, node.js	C++ et JavaScript	BSD	oui	`code.google.com/p/v8`	E5.1	
SpiderMonkey	Firefox, SeaMonkey	C++	MPL			E5.1	
Duktape	NetSurf…	C	MIT	oui	`duktape.org`	E5/E5.1 et quelques composants de E6	1.4.0 (10 janvier 2016)
Chakra	1ER, Microsoft Edge		propriétaire/MIT		la version ChakraCore	E5.1	
V7		C	GPL v2.0	oui	`github.com/cesanta/v7`	JavaScript 5.1	

Chapitre 4

Discussion et direction de recherche

Dans ce chapitre, il est question de décrire les spécifications des différents navigateurs étudiés et d'expliquer les choix effectués pour le projet. On parcourt les différentes options possibles et les avantages et inconvénients de chacune.

4.1 Les options

Après avoir présenté les différents navigateurs et leurs moteurs de rendus dans le chapitre précédent. Une sélection des possibilités qui sont offértes pour un portage vers AmigaOS 3.x est décrite ici.
Pour la suite, il est possible, soit de choisir de compiler un navigateur au complet, soit de compiler un moteur de rendu ce qui pourra diriger les travaux par la suite, soit sélectionner chaque technologie séparément et concevoir un navigateur de toute pièces.

4.1.1 Les navigateurs

En se fiant aux parts de marché, les navigateurs comme Chrome, FireFox, Safari ou encore MSEdge sont les meilleurs candidats pour une compatibilité maximale avec les standards du

Web. Ils sont complets et offrent toutes les commodités demandées par les utilisateurs (extensions). Ils sont aussi constamment mis à jour et entretenus par des sociétés très actives.
NetSurf est actuellement l'un des meilleurs navigateurs sur les plateformes Amiga de nouvelle génération et sur plusieurs autres systèmes roulant sur des machines de faibles capacités. Il intègre un moteur de rendu léger et créé entièrement par son équipe de développement, donc bien adapté au monde Amiga. Il est aussi l'un des projets qui utilisent Duktape.
Les développeurs de NetSurf sont constamment en train de corriger leurs versions pour qu'elles soient compatibles avec les Amiga Classics.
Links est un candidat relativement abordable et pouvant offrir une navigation stable. Links est donc une option qui pourrait répondre à un besoin minimaliste, mais bien présent. Son portage présente aussi l'avantage être simple à effectuer. Il est aussi toujours maintenu par son groupe de développement (contrairement à ELinks, son embranchement décrite dans la section 3).
iBrowse est un navigateur très réputé du monde Amiga. Durant d'années, la communauté a utilisé ce navigateur comme l'option principale pour l'accès à l'Internet. Depuis 2006, son développement est arrêté et les clés de licence ne sont plus fournis depuis 2007.
OWB est un navigateur populaire sous les systèmes Amiga sur PowerPC. Il supporte le HTML 5 et le CSS 2.1 [partiellement CSS3] en utilisant la puissance de Webkit. Il atteint aussi un score de 100% au test Acid3 sur certaines plateformes.
Links et NetSurf ne parviennent pas encore à supporter le test Acid3.

4.1.2 Les moteurs de rendu

Comme pour les navigateurs, si l'on se réfère aux parts de marché, le choix le plus judicieux serait Webkit ou Blink. Ces deux moteurs de rendus sont les plus performants du marché et sont soutenus par des communautés très actives. Mais la complexité de porter ce type de logiciel est élevée.
Gecko est aussi une option du niveau des deux précédents. Le problème de Gecko est qu'il est en développement depuis sa sorti en 1998 et que son code a été rarement nettoyé ou alléger.
NetSurf [engine], ELusive, Dillo [engine] sont des bibliothèques développées conjointement

avec les navigateurs qui les intègrent. Ils ont été développés de zéro et sont très liés.
Pour Trident et EdgeHTML, les moteurs de Microsoft ne sont pas à code source libre et sont donc des options non valables.
Servo est un moteur autonome, peut gourmant en puissance et énergie, intègrant son propre moteur JavaScript et développé par la communauté Mozilla. Il reste tout de même un produit expérimental et est développé en Rust, un langage qui n'est pas commun et n'appartient pas à la chaine de compilation GCC. Il faudra donc, dans le cas de Servo, créer un compilateur spécifique pour Rust.

4.1.3 Les moteurs ECMAScript

Les moteurs JavaScript sont généralement des bibliothèques développées en parallèle des navigateurs et des moteurs de rendu HTML auxquels ils vont être liés. La plupart des moteurs décrits dans le chapitre précédent doivent soit être pris avec les logiciels auquel ils sont adaptés, soit être mis à niveau et intégrés dans un nouvel environnement.
JavaScriptCore, V8, et ChakraCore sont des moteurs ECMAScript très performants, mais ils restent très liés aux moteurs de rendu qui les utilisent. Ils sont aussi d'une grandeur assez importante et peuvent impliquer des complications à la compilation, vu leurs fortes dépendances.
SpiderMonkey est aussi un bon choix en termes de performance, mais Mozilla compte apporter des modifications à son moteur de rendus. SpiderMonkey, et Gecko sont maintenu depuis plusieurs années, ce qui en fait de gros programmes, ils doivent donc être refactorisés et allégés.
Duktape est une option qui est facile à intégrer à n'importe quel projet en C/C++. Il est aussi très bien documenté. Il léger et régulièrement maintenu par son équipe de développement. Il est aussi déjà utilisé par le projet NetSurf-Browser.
V7 est un moteur JavaScript extrêmement léger. Il tient simplement en deux fichiers et est donc facile à intégrer dans un projet.

4.2 Discussion

Après avoir passé en revue les différentes options qui s'offrent au projet, des choix doivent être faits pour le projet.
Les performances et avantages qu'offre Webkit étaient les objectifs à l'origine de ce projet. Il est donc intéressant de ne pas perdre de vue la possibilité de réaliser ces objectifs. Il est cependant à considérer que le problème lié au Big Endian que rencontre Webkit avec les architectures comme celle de Amiga reste préoccupant.
On peut toujours envisager de réaliser un mélange de technologies et d'utiliser un moteur de rendu comme Gecko ou Blink ou un navigateur comme NetSurf ou Dillo et y intégrer Duktape ou V7. Il faut cependant consacrer du temps de développement, d'intégration et de test pour s'assurer du bon fonctionnement d'une telle démarche.
iBrowse est une option inanvisagable, en vue du statut actuel du projet. De même que pour les navigateurs de Microsoft qui sont propriétaires.
OWB est, comme la première solution, jusqu'à l'instant handicapé par le problème de Endianess de WebKit. Il reste tout de même une alternative intéressante si le problème est amené à se résoudre.
Faire des tests sur le moteur de rendu Servo et analyser ses capacités et performances pourrait aider à prendre une décision en ce qui le concerne.
NetSurf et Links sont des options attrayantes qui offrent l'avantage d'être accessibles et d'avoir d'anciennes versions qui on pu être portées sur les plateformes Amiga.

4.3 Directions

Les choix réalisés ont été dirigés essentiellement par les restrictions liées aux systèmes d'exploitation AmigaOS 3.x et aux limitations de la compilation croisée, ainsi que par la propension des logiciels à être portés sur de telles plateformes et au degré de complexité d'un tel portage.
La limitation de temps pour réaliser ce projet a aussi été une condition importante dans la ligne de conduite de celui-ci.

4.3.1 Première compilation croisée

Pour commencer et principalement tester le compilateur, on a compiler un compilateur GCC et on l'a testé avec succès sur un "Hello World".
On a utilisé :

- binutils 2.14
- gcc 2.95.3
- gcc 3.4.0

4.3.2 Links

Ensuit pour tester le compilateur sur un plus gros programme, on a choisi de porter le navigateur Links. Ceci permet de se familiariser plus avec la compilation croisée, les bibliothèques et l'environnement requis, et à l'utilisation d'AmigaOS.
Links fonctionne bien sur AmigaOS mais il reste un navigateur textuel qui n'est pas fait pour afficher les tout le contenu d'un page Web. Il ne répond donc pas au attentes du projet, il a un support partiel du HTML4 et du JavaScript.

4.3.3 NetSurf

NetSurf représente un bon choix pour être porté sur AmigaOS 3.x. Le portage a été effectué sur d'anciennes versions de NetSurf. On a donc effectué la compilation sur les trois dernières releases du navigateur qui était disponible au cours de la réalisation du projet. C'est à dire :

1. Release 3.3 : 10 Mars 2015
2. Release 3.4 : 16 Fevrier 2016
3. Release 3.5 : 6 Avril 2016

La figure 4.1 représente une capture d'écran de l'exécution de NetSurf sur AmigaOS 3.
Lors de la compilation Duktape géneère des erreurs de compilation et a dû être désactiver suivant suivant les conseils du développeur principal de NetSurf sur AmigaOS Chris Young [11].

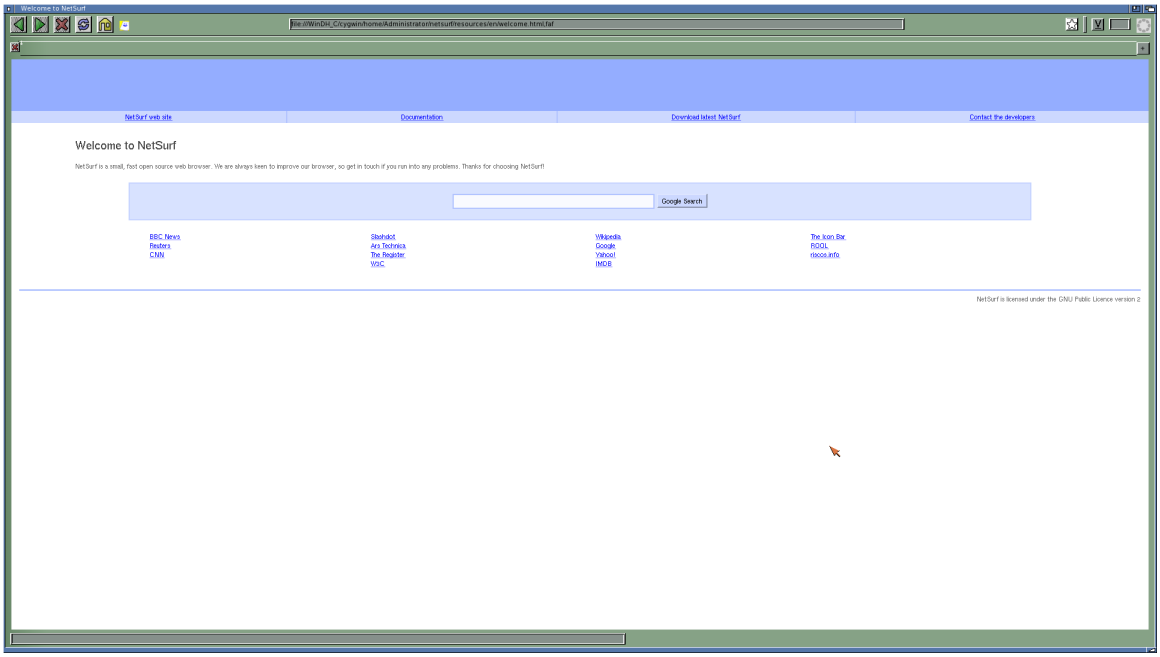

FIGURE 4.1 – NetSurf 3.5 sur AmigaOS 3

4.3.4 Webkit

Étant un moteur de rendu complet, WebKit semble un choix très propice au potage. Il offrirait un outil stable et compatible avec tous les standards. Le problème lié au Big Endian reste à résoudre.

La communauté autour de Odyssey Web Browser est en train de se pencher sur le sujet et essai de le résoudre pour sa version de WebKit. Une des directions qui peuvent être prise et de joindre leurs efforts et profiter de leur expérience pour mener à terme les objectifs.

Chapitre 5

Tutoriel

Dans ce chapitre, on explique comment procéder : pour créer le compilateur croisé pour Amiga 68k et pour préparer le workspace, ensuite les étapes de la compilation de NetSurf et de ses dépendences.
Ce tutoriel suit la documentation donnée par l'équipe développement de NetSurf [12]. Quelques modifications et précisions ont été apportées à cette documentation, car plusieurs erreurs de compilation pourraient survenir dans le cas contraire. On vous propose de regarder ces sources [5], [8] et [9] des tutoriels intéressants pour créer un compilateur croisé á partir de zéro.
Plus de précisions sur les éventuelles erreurs sont disponibles dans l'annexe 6.2.
Ce tutoriel comprend les étapes à suivre sous Window et sous Linux.

5.1 Préparation de l'éspace de travail

La première étapes est de créer le compilateur. On doit donc compiler le compilateur. Ce dernier sera brut, sans aucune bibliothèque. On devra compiler les bibliothèques adéquates par la suite pour le faire fonctionner.
Tout d'abord, il faut verifier que la bibliothèque epoch2 est bien accèssible. Pour cela, on exécute la commande suivant :

```
$ ls -al /usr/bin/autom4te*
$ lrwxrwxrwx 1 Administrator None     34 \\
/usr/bin/autom4te -> /usr/share/autotools/ac-wrapper.sh
$ lrwxrwxrwx 1 Administrator None     39 \\
/usr/bin/autom4te2.64 -> \\
/opt/gcc-tools/epoch2/bin/autom4te-2.64
$ -rwxr-xr-x 1 Administrator None 32181 \\
/usr/bin/autom4te-2.69
```

La commande étant la première ligne et le résultat les trois suivantes. Dans le cas où vous avez un résultat différent, vous devrez exécuter la commande qui suit, pour créer un lien symbolique sous Window :

```
$ ln -s /opt/gcc-tools/epoch2/bin/autom4te-2.64 \\
/usr/bin/autom4te2.64
```

et celles-ci sous Linux :

```
$ ln -s /usr/share/autotools/ac-wrapper.sh \\
/usr/bin/autom4te
$ ln -s /usr/bin/autom4te-2.69
```

Vous pouvez maintenant commencer. Il faut, en premier, télécharger le répertoire toolchains qui permet de mettre en place un environnement de compilation dédié à NetSurf-Browser. c'est à dire les binutils, les sources du compilateur GCC, ainsi que des bibliothèques liées au système vers lequel la compilation est prévue.

Les instructions suivantes vont télécharger toolchains et ses dépendences, puis décompresser les binutils et GCC spécifiques à Amiga OS 3.x, dit 68k.

```
$ git clone git://git.netsurf-browser.org/toolchains.git
$ cd toolchains/m68k-unknown-amigaos
$ make
```

Important : Durant l'exécution, des modifications sont à apporter à deux fichiers générés par l'exécution dans le dossier ~/toolchains/m68k-unknown-amigaos/gcc-3.4.6/gcc. Si ces modifications ne sont pas effectuées à temps, l'exécution échoue.

- dans le fichier c-parse.in : YYLEX et à remplacer par yylex() dans les lignes 638 et 2230
- dans le fichier collect2.c : modifiez la ligne 1594 par

```
redir_handle = open (redir, O_WRONLY | O_TRUNC | O_CREAT, 0666);
```

À ce stade, le compilateur GCC et les outils du gestion de binutils sont prêts.

5.2 Installation des dépendences

Vous devez maintenant ajouter le GCC et les binutils aux variables d'environnement du système. L'arborescence utilisée dans ce qui suit est celle par défaut, vous pouvez toujours en changer au besoin.

```
$ export PATH=/opt/netsurf/m68k-unknown-amigaos/cross/bin:$PATH
$ export PKG_CONFIG_PATH=/opt/netsurf/m68k-unknown-amigaos/env/
lib/pkgconfig/$PKG_CONFIG_PATH
$ export LD_LIBRARY_PATH=/opt/netsurf/m68k-unknown-amigaos/env/
lib:$LD_LIBRARY_PATH
```

Si tout de même vous avez changé l'arborescence, vous pouvez définir le PREFIX par défaut comme suit :

```
$ export $PREFIX=${HOME}/{INDIQUEZ_LE_CHEMIN_VOULU}
```

Vous devez, par la suite, préciser ce changement lors de la compilation des différentes bibliothèques.

L'étape qui suit, consiste à preparer le workspace où vas se dérouler la compilation.

```
$ cd toolchains/sdk
$ GCCSDK_INSTALL_CROSSBIN=/opt/netsurf/m68k-unknown-amigaos/cross/bin
GCCSDK_INSTALL_ENV=/opt/netsurf/m68k-unknown-amigaos/env make
```

5.3 La compilation

Vous êtes maintenant prêts à compiler les bibliothèques que requière NetSurf.

Pour toutes les commandes make qui seront exécutées par la suite, vous pourrez, selon le système d'exploitation ou la release de NetSurf, avoir besoin d'ajouter l'un des argumants suivants :

- TARGET=amigaos3
- BUILD=release
- PREFIX=/opt/netsurf/m68k-unknown-amigaos/env
- CC=m68k-unknown-amigaos-gcc
- HOST=m68k-unknown-amigaos

Il faut maintenant installer le buildsystem, comme suit :

buildsystem

```
$ git clone git://git.netsurf-browser.org/buildsystem.git
$ cd buildsystem
$ git pull
$ make HOST=m68k-unknown-amigaos clean
$ make HOST=m68k-unknown-amigaos
$ make HOST=m68k-unknown-amigaos install
$ cd ..
```

La présence de 3 fois la commande make est juste pour s'assurer du bon déroulement de l'exécution. Vous pouvez faire seulement le dernier :

- make clean : nettoie le dossier et supprimer les fichiers d'une éventuelle compilation ultérieure.
- make : vérifie si l'installation se déroulera correctement.
- make install : installe les binaires.

Dans ce qui suit, on installe les bibliothèques de dépendences de NetSurf. Elles ne sont pas toutes obligatoires. Il est conseillé de suivre l'ordre suivi par ce tutoriel pour ne pas rencontrer d'erreurs. Les erreurs possibles sont décrivent dans l'annexe p.III.

libwapcaplet, libparserutils, libcss, libhubbub et libdom sont les bibliothèques de base.

libnsbmp et libnsgif sont les principaux décodeurs d'images.

Et librosprite et libsvgtiny sont des décodeur optionnels.

libnsgif

```
$ git clone git://git.netsurf-browser.org/libnsgif.git
$ cd libnsgif
$ git pull
$ make clean
$ make
$ make install
$ cd ..
```

Libnsgif est une bibliothèque de décodage pour le format de fichier d'image GIF, écrit en C. Elle a été développée dans le cadre du projet NetSurf et est disponible pour une utilisation par d'autres logiciels sous la licence MIT [26].

libnsbmp

Libnsbmp est une bibliothèque de décodage pour BMP et ICO formats de fichier d'image [25].

```
$ git clone git://git.netsurf-browser.org/libnsbmp.git
$ cd libnsbmp
$ git pull
$ make clean
$ make
$ make install
$ cd ..
```

libwapcaplet [29]

```
$ git clone git://git.netsurf-browser.org/libwapcaplet.git
$ cd libwapcaplet
$ git pull
$ make clean
$ make
$ make install
$ cd ..
```

libparserutils

LibParserUtils est une bibliothèque pour la construction de parseurs efficaces [27].

```
$ git clone git://git.netsurf-browser.org/libparserutils.git
$ cd libparserutils
$ git pull
$ make clean
$ make
$ make install
$ cd ..
```

libhubbub

```
$ git clone git://git.netsurf-browser.org/libhubbub.git
$ cd libhubbub
$ git pull
$ rm -Rf examples
$ rm -Rf perf
$ make clean
$ make
$ make install
$ cd ..
```

Hubbub est une bibliothèque d'analyse conforme HTML5.

HTML5 définit un algorithme d'analyse, basée sur le comportement des navigateurs grand public, qui fournit des instructions sur la façon d'analyser toutes les balises, à la fois valides et non valides.

libdom

LibDOM est une mise en œuvre du DOM W3C [24].

```
$ git clone git://git.netsurf-browser.org/libdom.git
$ cd libdom
$ git pull
$ rm -Rf examples
$ make clean
$ make
$ make install
$ cd ..
```

libcss

Libcss est un moteur CSS (Cascading Style Sheet) pour l'analyse et la sélection [23].

```
$ git clone git://git.netsurf-browser.org/libcss.git
$ cd libcss
$ git pull
$ make clean
$ make
$ make install
$ cd ..
```

libsvgtiny

```
$ git clone git://git.netsurf-browser.org/libsvgtiny.git
$ cd libsvgtiny
$ git pull
$ make clean
$ make
$ make install
$ cd ..
```

Libsvgtiny est une implmentation de SVG Tiny [28].
L'idée générale de la bibliothèque est de prendre un certain SVG comme entrée, et le retour une liste des chemins et des textes qui peuvent être rendus facilement. Elle ne fait pas le rendu réel.

libnsutils

```
$ git clone git://git.netsurf-browser.org/libnsutils.git
$ cd libnsutils
$ git pull
$ make clean
$ make
$ make install
$ cd ..
```

libutf8proc

Libutf8proc est une bibliothèque d'encodage en UTF-8.

```
$ git clone git://git.netsurf-browser.org/libutf8proc.git
$ cd libutf8proc
$ git pull
$ make clean
$ make
$ make install
$ cd ..
```

nsgenbind

Nsgendind est une bibliothèque qui s'occupe de la génération des *bindings* du JavaSrcipt.

```
$ git clone git://git.netsurf-browser.org/nsgenbind.git
$ cd nsgenbind
$ git pull
$ make clean
$ make
$ make install
$ cd ..
```

Construction de NetSurf

Pour finir, vous devez compiler le navigateur. L'ajout de l'argument TARGET est obligatoire ici.

```
$ git clone git://git.netsurf-browser.org/netsurf.git
$ cd netsurf/
$ git pull
$ make TARGET=amigaos3 clean
$ make TARGET=amigaos3
```

En suivant ce tutoriel, étapes par étapes, le navigateur NetSurf devrait compilé sans erreurs. On rappel que si tout de même vous en avez, l'annexe contient toutes les erreurs rencontrées durant ce projet et comment les corriger.

Chapitre 6

Travaux liés et futurs

Après avoir réussi à compiler les navigateurs Links et NetSurf vers AmigaOS 3.x, il semble que d'autres travaux pourraient encore être effectués sur le sujet et élargir les horizons. On décrit ici les travaux dirigés par la communauté en ce moment et les travaux qui pourraient être fait suite à celui-ci.

6.1 Travaux liés

Durant les recherches et à travers les lectures sur différents forums dédiés à l'univers Amiga [6] [30], on a remarqué qu'il existe des projets portant sur le sujet traité. Plusieurs utilisateurs ou groupes d'utilisateurs se sont mis en quête d'un navigateur Web qui pourrait fonctionner sur AmigaOS 3.x avec le plus possible de compatibilité avec les standards actuels de l'Internet ou tout simplement un navigateur stable qui leur permettraient d'y accéder [38].
Certain de ces travaux étaient réalisés en solo par un fan ayant des connaissances en développement. Il réalise un portage puis le partage avec la communauté sur les forums. Par exemple, un utilisateur du forum Amiga.net a partagé son navigateur Links [36].
D'autres projets, menés en groupe, se sont consacrés à NetSurf et on réussi des portages du navigateur dans ses anciennes versions jusqu'à la 3.3. Le résultat de leurs projets sont partagés sur Github sur les références suivantes [32] et [33].
L'équipe de développement de Odyssey Web Browser est aussi à la recherche de développeurs

qui pourrait l'aider à compléter le portage de son navigateur sur AmigaOS 3.x. durant le projet, la communauté a publié des offres pour trouver une solution au problème de Endianess que rencontre WebKit sur la plateforme Amiga.
Dernièrement, l'équipe de Odyssey a publié son site Web [35] que le problème a bien été résolu est que OWB est maintenant apte à être porté sur AmigaOS 3.x.

6.2 Travaux futurs

Le travail réalisé ouvre la porte a d'autres essais de portage et de travaux à venir :
On peut imaginer essayer de créer un compilateur croise pour le langage Rust. De tels compilateurs existent déjà pour ce langage, notamment vers les micro-ordinateurs Raspberry PI. Ensuite, il serait possible de réaliser le port du moteur de rendu Servo.
Vu l'incapacité de faire fonctionner Duktape pour l'instant, on peut aussi envisager d'intégrer V7 à NetSurf, puis de refaire la compilation. L'environnement étant déjà prêt, cela simplifie cette tâche. Ou encore de résoudre les problèmes lié à Duktape.
Un autre travail intéressent serait d'approfondir la faisabilité de compiler WebKit. Ce dernier apporterait une qualité supérieure à la navigation sur Amiga. Ce but étant peut-être presque atteint d'après les travaux de Odyssey Web Browser.

Conclusion

Dans ce projet, il a été question de trouver un moyen d'améliorer le quotidien des utilisateurs des Amiga et AmigaOS 3.x émulés en essayant d'offrir un moyen de naviguer sur l'Internet à l'aide d'un navigateur Web pouvant supporter les standards actuels du Net.
Ce travail offre à la communauté Amiga une feuille de route qui peut être une référence pour diriger d'éventuel portage de navigateurs ou autres programmes vers les systèmes Amiga de troisième génération. C'est une ouverture sur d'autres travaux et de nouvelles perspectives.
Tout d'abord, ce document contient un historique de l'épopée Amiga depuis sa création à son statut actuel. Ce bref retour vers les années 80 montre l'important rôle qu'a joué Amiga dans l'informatique et peut être une source de découverte et d'intérêt pour la nouvelle génération qui ignore l'existence de cette page de l'histoire. Il donne un aperçu sur l'état actuel du marché des navigateurs Web et de la multitude de technologies qui les entours. Enfin, ce document contient un tutoriel qui explique en détail comment créer son propre environnement de compilation croisée pour AmigaOS 3.x et peut servir comme tutoriel pour d'autres types de compilations croisées.
De plus, ce travail offre à la communauté un environnement de compilation prêt à l'emploi, qui a était testé et est fonctionnel. Il pourra être mis à disposition d'utilisateur sur une plateforme de partage. Le NetSurf compilé l'étant partagé sur Github [34].

Glossaire

Acid3	un test proposé par W3C pour valider leur conformité aux standards de rendu HTML. Ce test est noté sur 100 points.
DOM	Document Object Model
DOS	Disk Operating System
GCC	GNU Compiler Collection
GNU	GNU's Not UNIX
HTML	HyperText Markup Language
IDE	Integrated Development Environment
KJS	KDE JavaScript
KHTML	Konqueror HTML, le moteur de rendu du navigateur web du projet KDE
Licences : -BSD -GPL -MIT -MPL	 Berkeley Software Distribution License GNU General Public License Massachusetts Institute of Technology License Mozilla Public License
NCSA	National Center for Supercomputing Applications
PDA	Personal Digital Assistant
RISC	Reduced Instruction Set Computer
UAE	Unix Amiga Emulator
UTF-8	Universal Coded Character Set + Transformation Format - 8-bit
W3C	World Wide Web Consortium

Bibliographie

[1] Amidevapp. http://amidevcpp.amiga-world.de/. visité le : 2016-01-21.

[2] Amiga 30. http://amiga30.com/. visité le : 2016-04-14.

[3] Amiga 30. http://www.amiga30.eu/. visité le : 2016-04-14.

[4] Amiga history. http://www.amigahistory.co.uk/ahistory.html. visité le : 2016-05-02.

[5] Build amiga gcc 3.3.3 cross compiler for linux or windows. https://www.libsdl.org/extras/amigaos/cross/src/article.html. visité le : 2016-02-06.

[6] Community. http://www.amigaos.net/content/5/community. visité le : 2016-02-28.

[7] Cross compile. https://shadoware.org/post/cross-compilation-compiler-un-programme-pour-ms-windows-sous-gnu- visité le : 2016-02-27.

[8] Cross-compiling. http://www.chingu.asia/wiki/index.php?title=Cross-compiling&from=Cartoon7. visité le : 2016-02-07.

[9] Cross-compiling for amigaos 3.x. http://kas1e.mikendezign.com/zerohero_crosscompilers_backup/cross-compiler_os3.html. visité le : 2016-02-06.

[10] Dillo. https://fr.wikipedia.org/wiki/Dillo. visité le : 2016-02-03.

[11] duktape error netsurf os3. http://www.amiga.org/forums/showthread.php?t=70612. visité le : 2016-03-14.

[12] Getting coding. http://wiki.netsurf-browser.org/Documentation/GettingCoding. visité le : 2016-01-31.

[13] Histoire : 1997, premier objectif : supporter la communauté amiga existante. http://www.amigaimpact.org/articles/histoire-1997-premier-objectif-supporter-la-communaute-amiga-exis visité le : 2016-04-24.

[14] Histoire de l'amiga - anne 1980 et avant. http://obligement.free.fr/articles/amiga_histoire_1980.php. visité le : 2016-01-24.

[15] Histoire de l'amiga - anne 1982. http://obligement.free.fr/articles/amiga_histoire_1982.php. visité le : 2016-03-25.

[16] Histoire de l'amiga - anne 1983. http://obligement.free.fr/articles/amiga_histoire_1983.php. visité le : 2016-03-25.

[17] Histoire de l'amiga - anne 1984. http://obligement.free.fr/articles/amiga_histoire_1984.php. visité le : 2016-03-25.

[18] Histoire de l'amiga - anne 1985. http://www.obligement.free.fr/articles/amiga_histoire_1985.php. visité le : 2016-03-25.

[19] Histoire de l'amiga - anne 1988. http://obligement.free.fr/articles/amiga_histoire_1988.php. visité le : 2016-01-24.

[20] Histoire de l'amiga - anne 1994. http://obligement.free.fr/articles/amiga_histoire_1994.php. visité le : 2016-01-24.

[21] Histoire de l'amiga - anne 1996. http://obligement.free.fr/articles/amiga_histoire_1996.php. visité le : 2016-01-24.

[22] la scission de la communauté amiga. http://obligement.free.fr/articles/scissioncommunaute.php. visité le : 2016-04-29.

[23] Libcss. http://www.netsurf-browser.org/projects/libcss/. visité le : 2016-01-18.

[24] Libdom. http://www.netsurf-browser.org/projects/libdom/. visité le : 2016-01-18.

[25] Libnsbmp. http://www.netsurf-browser.org/projects/libnsbmp/. visité le : 2016-01-18.

[26] Libnsgif. http://www.netsurf-browser.org/projects/libnsgif/. visité le : 2016-01-18.

[27] Libparserutils. http://www.netsurf-browser.org/projects/libparserutils/. visité le : 2016-01-18.

[28] Libsvgtiny. http://www.netsurf-browser.org/projects/libsvgtiny/. visité le : 2016-01-18.

[29] Libwapcaplet. http://www.netsurf-browser.org/projects/libwapcaplet/. visité le : 2016-01-18.

[30] liens. http://amigamuseum.emu-france.info/liens.htm. visité le : 2016-04-30.

[31] Mieux connaitre ses outils n1 : le moteur de rendu html. http://www.ux-republic.com/mieux-connaitre-ses-outils-n1-le-moteur-de-rendu- visité le : 2016-04-04.

[32] Netsurf-68k. https://github.com/arczi84/NetSurf-68k. visité le : 2016-03-03.

[33] Netsurf os3. https://github.com/kyllikki/netsurf. visité le : 2016-03-11.

[34] Netsurf-os3. https://github.com/EyMenZ/NetSurf-OS3. visité le : 2016-05-09.

[35] Odyssey web browser. https://www.power2people.org/projects/odyssey/. visité le : 2016-05-10.

[36] Recompilingporting links. http://www.chingu.asia/wiki/index.php?title=Recompilingporting+Links#TOC_2. visité le : 2016-02-24.

[37] La sam440ep. http://www.amiga-ng.org/sections.php?op=viewarticle&artid=52. visité le : 2016-05-05.

[38] We need an ibrowse replacement for 68k!!! http://www.amiga.org/forums/archive/index.php/t-63990.html. visité le : 2016-01-24.

Annexe

Cette annexe recense les erreurs rencontrer durant les tests de compilation de NetSurf, ainsi que la façon de les résoudre. Les étapes décritent dans le chapitre 5 ne seront pas reprises ici. Ceci est un complément pour le tutoriel.

Important : si un erreur survient, vous devrez exécuter un make clean avant de recompiler la bibliothèque.

Pour commencer, tapez la commande qui suit peut être très instructif. Elle nous informe sur le système cible du compilateur. Donc si la réponse est comme celle de l'exemple, c'est-à-dire que le compilateur donne un résultat pour AmigaOS 3.x et que le compilateur est bien dans les variables d'environnement du système $PATH.

```
$ m68k-unknown-amigaos-gcc -dumpmachine
m68k-unknown-amigaos
```

Erreurs liées au symbolink

première étape du tutoriel 5.1 et 5.1.

```
$ Makefile:109 : la recette pour la cible \\
/usr/local/home/toolchains/m68k-unknown-amigaos/ \\
builddir/build-steps/bootstrap-compiler.d a \'echou\'ee
$ make: *** [/usr/local/home/toolchains/ \\
m68k-unknown-amigaos/builddir/build-steps/ \\
bootstrap-compiler.d] Erreur 2
```

Ou

```
$ Makefile:126 : la recette pour la cible \\
/home/Administrator/toolchains/m68k-unknown-amigaos/ \\
builddir/build-steps/srcdir-step3.d a \'echou\'ee
$ make: *** [/home/Administrator/toolchains/ \\
m68k-unknown-amigaos/builddir/build-steps/srcdir-step3.d] Erreur 1
```

Ou

```
$ Makefile:155 : la recette pour la cible \\
/home/Administrator/toolchains/m68k-unknown-amigaos/ \\
builddir/build-steps/binutils.d a \'echou\'ee
$ make: *** [/home/Administrator/toolchains/ \\
m68k-unknown-amigaos/builddir/build-steps/binutils.d] Erreur 1
```

Ces erreurs signifient que vous avez : soit oublier de télécharger epoch2 ou autoconf dans l'étape de préparation de l'environnement, soit oublier l'étape de création du lien symbolique 5.1.

gcc introuvable ou erroné

Le cas suivant survient quand un problème arrive pendant le téléchargement de gcc par toolchains. Il pointe alors dans un mauvais emplacement. On conseille dans ce cas de rééffectuer l'opération depuis le debut.

```
$ Makefile:78  : la recette pour la cible      \\
/usr/local/home/toolchains/m68k-unknown-amigaos \\
/builddir/build-steps/clib2.d      a \'echou\'ee
$ make: *** [/usr/local/home/toolchains/m68k-unknown-amigaos \\
/builddir/build-steps/clib2.d] Erreur 2
```

Erreur liée à la compilation des bibliothèques

Cette erreur peut arriver seulement sous Windows, le compilateur confond alors le chemin /opt/netsurf/m68k-unknown-amigaos/cross/bin/m68k-unknown-amigaos-.. avec /opt/netsurf/m68k/unknow

```
$ make: *** [build-i686-pc-cygwin-m68k-unknown-amigaos-release-lib-sta
/NOM_DE_LA8BIBLIOTHEQUE.a] Erreur 127
```

La solution est soit de créer ce chemin en copiant le comtenu du premier dans le nouveau, soit de modifier le Makefile.
Si l'erreur est Erreur 1 : Fatal, cela veut dire que c'est une erreur de dépendence, et que un bibliothèque a é'té compilée avant une dont elle dépend.

Oublier l'export

Les cas suivants surviennent lorsque vous oublier de réaliser l'étape 5.2, soit en entier, soit partiellement.

```
$ Makefile:332 : la recette pour la cible \\
/home/Administrator/toolchains/sdk/builddir-m68k-unknown-amigaos \\
/build-steps/libpbl.d a \'echou\'ee
$ make: *** [/home/Administrator/toolchains/sdk/ \\
builddir-m68k-unknown-amigaos/build-steps/libpbl.d] Erreur 2
```

```
$ make[1] : on quitte le r\'epertoire /home/Administrator \\
/toolchains/sdk/builddir-m68k-unknown-amigaos/libjpeg/jpeg-8 d
$ touch /home/Administrator/toolchains/sdk/ \\
builddir-m68k-unknown-amigaos/build-steps/libjpeg.d
```

Ici, il pointe sur le mauvais gcc.

```
$ Makefile:797 : la recette pour la cible \\
build-CYGWIN_NT-6_3_WOW-amigaos3/amiga_os3support.o \\
a \'echou\'ee
$ make: *** [build-CYGWIN_NT-6_3_WOW-amigaos3/ \\
amiga_os3support.o] Erreur 1
```

Ici l'export de $PKG_CONFIG_PATH.

```
$ Makefile:797 : la recette pour la cible \\
build-CYGWIN_NT-6.3_WOW-amigaos3/javascript_duktape_dukky.o \\
a \'echou\'ee
$ make: *** [build-CYGWIN_NT-6.3_WOW-amigaos3/ \\
javascript_duktape_dukky.o] Erreur 1
```

```
$ Makefile:187 : la recette pour la cible \\
/home/Administrator/toolchains/sdk/builddir-m68k-unknown-amigaos \\
/build-steps/libiconv.d a \'echou\'ee
$ make: *** [/home/Administrator/toolchains/sdk/ \\
builddir-m68k-unknown-amigaos/build-steps/libiconv.d] Erreur 2
```

Portage de Duktape

L'erreur qui suit est causée par l'impossiblilté votre compilateur à compiler Duktape. La solution est de créer un fichier Makefile.config sous le dossier netsurf en suivant les instructions de Makefile.default (et non pas en le copiant).

```
$ Makefile:797 : la recette pour la cible \\
build-CYGWIN_NT-6.3_WOW-amigaos3/javascript_dukky.o \\
a \'echou\'ee
$ make: *** [build-CYGWIN_NT-6.3_WOW-amigaos3/ \\
javascript_dukky.o] Erreur 1
```