

SOMMAIRE

<u>1. OBJET</u>	2
<u>2. INTRODUCTION</u>	2
<u>3. PRINCIPE D'INTEGRATION</u>	2
<u>4. PROCEDE DE GENERATION DE LA VERSION 1.0a</u>	3
4.1 Contenu des répertoires.....	3
4.2 Configuration utilisée.....	3
4.3 Opérations de compilation	4
4.4 Création de la version.....	4
<u>5. LISTE DES VERSIONS ET EVOLUTIONS LIEES AUX DIFFERENTES VERSIONS DE TRAVAIL</u>	5

Indice	Date de diffusion	Intitulé
A	02/06/1997	Document initial

Toute modification de ce document par rapport à sa version précédente est repérée par un trait vertical dans la marge gauche.

1. OBJET :

Objet : Ce document constitue le Dossier d'Intégration du logiciel OTHO dans le cadre du projet OTHELLO.

Domaine d'application : Logiciel destiné à une application type grand public.

Documents de référence : Cahier des Charges indice B du 22/03/1997.
Spécification Technique de Besoin indice A du 22/03/1997.
Document de Conception Préliminaire indice A du 26/04/1997.
Document de Conception Détaillée indice A du 02/06/1997
Compte-Rendu de Tests Unitaires indice A du 02/06/1997

2. INTRODUCTION :

Ce document a pour but d'expliquer et justifier pourquoi nous n'avons pas effectué de tests d'intégration pour la réalisation des versions de notre logiciel OTHO. De plus, il explicite le procédé de génération de la version 1.0a d'OTHO, dont le principe a été appliqué à toutes les versions antérieures et sera appliqué pour toutes les versions à venir.

3. PRINCIPE D'INTEGRATION :

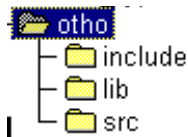
Dans notre cas, les tests d'intégration se sont révélés inutiles, du fait de notre principe de programmation. En effet, nous avons choisi un principe de programmation itératif : à chaque fois qu'un module, ou qu'un groupe de modules, est créé ou modifié, c'est une nouvelle version d'OTHO qui est créée.

Ce principe s'est directement imposé à nous-même. Il y a plusieurs raisons à cela :

- ❖ Tout d'abord, grâce à l'utilisation de la maquette comme base de la programmation, nous sommes partis d'un existant qu'il fallait modifier. Le codage de notre application s'est donc déroulé comme une programmation d'évolutions majeures sur un logiciel déjà existant.
- ② Existence d'une forte interactivité des modules entre eux. Notre logiciel est composé de plusieurs modules qui sont tous liés entre eux. Ainsi, il nous était impossible de tester séparément les fonctions du logiciel étant donné le nombre important d'appels de fonctions réalisés (voir le graphe d'appel en annexe du DCD Ind A.).
- ③ Afin d'avoir un principe de programmation simple : les modules étaient créés au fur et à mesure puis intégrés par paquets en une version et en un seul exécutable pour réaliser ou modifier un service important du logiciel. L'exécutable **otho.exe** issu de la compilation ne représente qu'un seul fichier ayant une taille de 200 ko, ce qui est relativement faible.
- ④ De plus afin de réaliser les tests unitaires, il nous fallait absolument compiler nos sources : comment faire tourner seul un de nos modules ?! On aurait pu compiler nos modules séparément mais, étant donné la forte interactivité (voir point n° 2), quelles procédures aurions nous du compiler, et lesquelles pas ? En outre, quel aurait été l'intérêt d'une telle méthode pour un logiciel comme OTHO ? Cela nous aurait fait perdre du temps (précieux !!) et nous aurions augmenté le risque d'erreur, ce qui aurait été stupide de notre part.
Ainsi, le choix d'une programmation itérative nous a permis de réaliser indirectement les tests d'intégration dans nos tests unitaires.
- ⑤ Aucun autre exécutable n'est nécessaire à OTHO pour accomplir son fonctionnement : il fonctionne donc seul, sans aide de logiciels extérieurs : quel est l'intérêt alors de tester son intégration externe ??

4. PROCEDURE DE GENERATION DE LA VERSION 1.0a :

4.1 Contenu des répertoires :



Fichiers « header » (*.h)
 Librairie utilisée par OTHO (svgacc.lib)
 Fichiers sources



- otho				
include				
lib				
src				
capture.h				
couleur.h				
dialogue.h				
gifsave.h				
global.h				
keyboard.h				
menupal.h				
prototyp.h				
svgacc.h				
type.h				
variable.h				
version.h				
readme.1st				
svgadem1.c				
svgadem2.c				
svgademo.c				
history.cc				
file_id.diz				
svgademo.exe				
dragon.fnt				
order.frm				
uttower.gif				
svgacc.h				
svgademo.h				
svgacc.lib				
packing.lst				
desc.sdi				
svgacc21.txt				
2joueur.c	14923	21/05/1997	23:08:12	
algorithm.c	38368	21/05/1997	22:53:44	
apropos.c	1959	27/04/1997	14:35:50	
capture.c	3754	25/05/1994	11:12:18	
dialogue.c	45081	21/05/1997	23:01:08	
ecran1.c	2798	27/04/1997	14:36:36	
fichier.c	5588	21/05/1997	22:09:56	
gifsave.c	31590	25/05/1994	09:59:50	
initiat.c	3635	21/05/1997	22:04:28	
jouer.c	42531	21/05/1997	23:12:46	
keyboard.c	1321	15/12/1996	16:36:28	
materiel.c	6401	27/04/1997	14:36:48	
memoire.c	5702	27/04/1997	14:37:00	
menupal.c	9872	25/05/1994	10:59:08	
ordinat.c	6903	21/05/1997	22:06:04	
otho.c	3448	25/05/1994	09:53:32	
ouvert.c	28290	21/05/1997	23:04:14	
regles.c	13409	19/05/1997	18:13:20	
scores.c	13643	21/05/1997	21:36:36	
services.c	3968	14/01/1997	00:21:42	
souris.c	9183	25/05/1994	10:59:30	
statist.c	8637	21/05/1997	23:29:28	
variable.c	2920	18/05/1997	16:10:06	
video.c	9400	25/05/1994	11:14:52	
visual.c	4154	19/05/1997	18:22:08	

4.2 Configuration utilisée :

Matériel : Cyrix P 166+ 133 Mhz 24 Mo RAM
Logiciel : Borland Turbo C ++ Version 3.0

4.3 Opérations de compilation :

Voici la liste exhaustive des opérations à suivre pour générer l'exécutable :

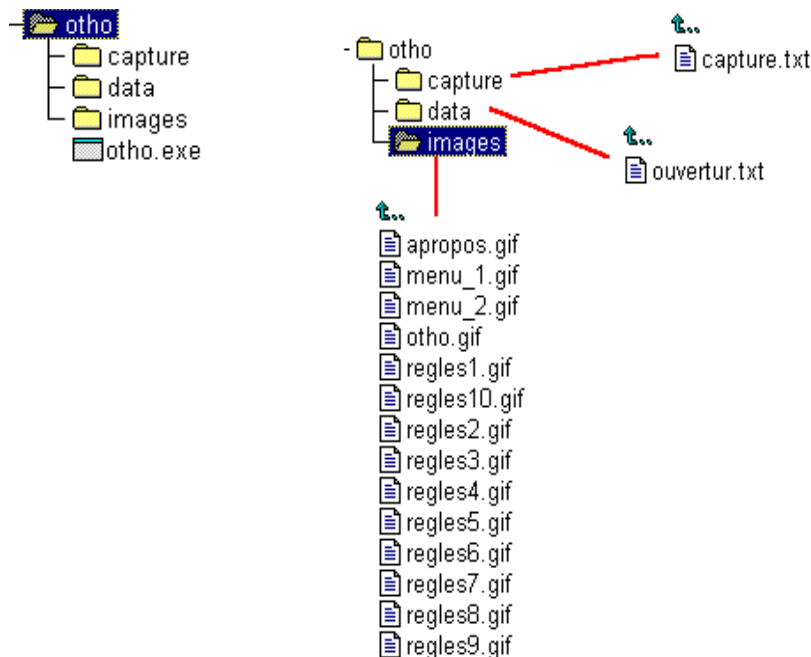
- Se placer dans le répertoire « otho »
- Lancer Turbo C en appelant la commande TC (l'accès sur le disque est précisé préalablement dans le PATH de l'autoexec.bat)
- Ouvrir le fichier **projet\otho.prj** s'il n'est pas déjà ouvert
- Compiler en appuyant sur F9 (ou en utilisant le menu « compiler »)
- Génération d'un programme exécutable sous le répertoire « otho » s'appelant **otho.exe**

Mais la version générée n'est pas encore complète : pour pouvoir fonctionner, il est nécessaire qu'OTHO.exe soit associé aux fichiers d'images, au format **.gif**, et au fichier d'ouvertures, au format **.txt**.

4.4 Création de la version :

Association de l'exécutable **otho.exe** avec les répertoires « images » et « data » comportant les fichiers nommés dans le schéma ci-dessous. Pour la spécificité de la version 1.0a qui permet de faire des captures d'écran (version β oblige !!), il est nécessaire d'avoir le répertoire « capture » :

Soit, en final :



OTHO version 1.0a est créé : il n'y a plus qu'à l'utiliser !!

5. LISTE DES VERSIONS ET EVOLUTIONS LIEES AUX DIFFERENTES VERSIONS DE TRAVAIL :

Le contenu de ce paragraphe est principalement issu du fichier **version.h**, administré par Patrick DEIBER, et permettant de gérer la traçabilité de toutes les évolutions du logiciel OTHO :

DATE	VERSION	DESCRIPTION
22/03/1997	v 0.4	<u>Maquette</u> : image d'accueil jeu à deux joueurs humains jeu contre l'ordinateur (profondeur 3 : algorithme min-max)
Diverses dates	v 0.5 à v 0.6b	Diverses évolutions d'algorithme de recherche Finalisation de l'interface Homme/machine et des écrans.
06/04/1997	v0.7a	Création du fichier version.h
08/04/1997	v0.7b	Amélioration de la gestion des couleurs
16/04/1997	v0.7c	Statistiques du jeu
18/04/1997	v0.7d	Amélioration de l'ecran de jeu et du niveau de l'ordinateur en fin de jeu
20/04/1997	v0.8a	Mise en place des boutons précédent, suivant, et conseil du jeu. Mise en place du choix initiation du menu général
27/04/1997	v0.9a	Amelioration de la gestion de l'affichage des statistiques et des dialogues Mise en place du dialogue de saisie Saisie des noms des joueurs et du nom de la partie a reprendre
02/05/1997	v0.9b	Mise en place du choix règles du jeu
06/05/1997	v0.9c	Mise en place de la sauvegarde et du chargement d'une partie
06/05/1997	v0.9d	Amelioration de la lecture du code
08/08/1997	v0.9e	Mise a jour du nombre d'ecrans pour les règles du jeu
18/05/1997	v0.9f	Mise en place des high scores
19/05/1997	v0.9g	Mise en place des ouvertures
20/05/1997	v0.9h	Suite de la mise en place des ouvertures Modification presentation high score Suppression de la sauvegarde pour le niveau initiation
21/05/1997	v1.0a	Préparation de la version β du jeu Otho

Toutes ces versions de travail sont archivées (exécutables + sources) sous le format **.zip**.