

Amiga Report Technical Journal

Gregory R. <Winter@Oubliette.com>

COLLABORATORS

	TITLE : Amiga Report Technical Journal		
ACTION	NAME	DATE	SIGNATURE
WRITTEN BY	Gregory R. <Win- ter@Oubliette.com>	August 10, 2024	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

1	Amiga Report Technical Journal	1
1.1	Amiga Report Technical Journal, Volume 1, Issue 1, 04/07/95	1
1.2	ARTJ - Volume 1, Issue 1 - Table of Contents	1
1.3	Editorial: Amigas in the Mist	2
1.4	Liftoff!, by Jason Compton	4
1.5	Programming in Zen	4
1.6	Letters to the Editor	5
1.7	Pointers -- Hot spots, news, and files	5
1.8	The Man from M.Y.S.T.I.K.	6
1.9	Conference: The Amiga-OS Project	14
1.10	The Heart of Installer	43
1.11	Methods to the Madness	45
1.12	Revision Control and SMAKE under SAS/C	50
1.13	smakefile	55
1.14	A	55

Chapter 1

Amiga Report Technical Journal

1.1 Amiga Report Technical Journal, Volume 1, Issue 1, 04/07/95

Amiga Report Technical Journal and ARTJ are Copyright ©1995, FS Publications, All Rights Reserved.

Producing Editor, Gregory R. Block, Winter@Oubliette.com, Oubliette Software.
Supervising Editor, Jason Compton, jcompton@xnet.com, FS Publications.

Views, opinions, and articles represented herein are not necessarily those of the editors and staff of the Amiga Report Technical Journal, hereafter ARTJ, or of FS Publications. Permission to reprint articles is hereby granted, unless otherwise noted. Reprints must, without exception, include the name of the publication, the date and issue number, and the author's name. ARTJ and/or portions therein may not be edited in any way without prior written permission. However translation into a language other than English is acceptable, provided the editor is notified beforehand and the original meaning is not altered. ARTJ may be distributed on privately owned, non-profit bulletin board systems (fees to cover cost of operation are acceptable), and major online services, such as, but not limited to, Delphi, Portal, CompuServe, America Online, and BIX. Distribution on public domain disks is acceptable provided proceeds are only to cover the cost of the disk (e.g., no more than \$5 US). Distribution on for-profit magazine cover disks requires written permission from the editor. ARTJ is a non-profit publication written by and for the Amiga development community out of respect and love of its members. ARTJ, its staff, and its contributors are not and cannot be held responsible for the use or misuse of information contained herein, or the results obtained therefrom. ARTJ is not affiliated with Commodore-Amiga, Inc., Commodore Business Machines, Ltd., or any other publication, in any way. All items quoted in whole or in part are done so under the Fair Use Provision of the Copyright Laws of the United States Penal Code. Any Electronic Mail sent to the editors may be reprinted, in whole or in part, without any previous permission of the author, unless said electronic mail is specifically requested not to be reprinted.

1.2 ARTJ - Volume 1, Issue 1 - Table of Contents

Departments

Editorial: Amigas in the Mist
Liftoff!, by Jason Compton
Programming in Zen
Letters to the Editor
Pointers -- Hot spots, news, and files

Interviews

The Man from M.Y.S.T.I.K.

Conferences

The Amiga-OS Project

Features

RobR takes us into the Heart of Installer, part one in the series.
Methods to the Madness, a guide to BOOPSI authoring, from Chris Aldi.
Revision Control and SMAKE under SAS/C by Osma Ahvenlampi, with an example.
Church Windows, reprinted without permission from
comp.sys.amiga.advocacy.

1.3 Editorial: Amigas in the Mist

Amigas in the Mist
by Gregory R. Block

I wonder, sometimes, at what it means to be a developer; what makes me different from the many numbers of people who exist, mostly, satisfied with what they have. For some reason, something drives me to write applications, and even to write here, and I believe that something is the something that drives all developers; the need to create; the need to leave something behind that is, in some way, a memory, a continuation, a belief, a thought, anything. Writers who keep memoirs use their memoirs to remind themselves of who they are, but also to know that who they are was someone, and is someone, too; their memories are more than just imaginary thoughts, they're physical, sitting in the log they keep. Journalists stash their articles. Prolific authors, or even not-so-prolific-but-well-intentioned authors like Dale Larson, go out on book signings; something that satisfies the ego, the publisher, and the audience, all without earning a death threat (unless you're a Green Card Lawyer). And just like most people, engineers, both software and hardware alike, feel that same need; witness the signed A1000 cases of so long ago, the secret messages hidden in the Amiga's OS.

See, I, as a man, and particularly as a gay man, if I were to psychoanalyze for a bit, am missing a rather large portion of the average person's life -- I have no means of procreation, I have, and will likely never have, a child.

A child is more than just a product; it is something you leave behind, a walking, talking legacy that carries with it, whether it be good or bad, an impression of you, a small image of the person you are. Developing software or hardware is more than just creation; it is the leaving behind, in some way, some small piece of yourself for others to see. It is as much a gift to the users as it is a gift to yourself, some temporary freedom from

that parenting instinct, or just a freedom in knowing that something you have created or done will affect someone in profound and unseeable ways; in software and hardware, many someones.

So, it is unsurprising that I, in my youth, turned towards computers when my social life went downhill; years of blowing grading curves and being the school "nerd" have made me a cynic; years of living through my computer have made me a developer, and given me the ideas and vision I need to keep my faith, to perpetuate my conception of the way things should be going; it is these "visionary" ideas I hold that makes me continue to be a developer once the product is done, just as it is everyone's. It's the feeling that what you do, the work you provide, is more than just a "stopgap", or hole filling; it's the feeling that the software you've developed is somehow of great importance, something that could really change the way things are, the way things work, the way people think. It's that "Coke and a smile" attitude, that feeling of the machine beneath your fingers, that keeps many going.

It's certainly what keeps me here.

Here, of course, meaning the Amiga; which, arguably, has fallen well behind in its ability to run modern applications due to its lack of hardware and software services. I once was one of the lead arguers in the belief that there was nothing wrong with the approach the Amiga's OS was taking; but the more entwined you become in the developer community, the larger the application you're dealing with, the more you realize that you're fighting a very special uphill battle, one that not everyone, on every platform, needs to fight.

Which of course leads me back to that aromatic, rotting corpse we've come to know as Commodore International, and the hope that some form of life feasts on it. Personal hopes, of course, are for CEI, whose ideas seem to be more in line with a modern attempt at building a company capable of competing as something other than a video game system. Regardless, one MUST require of whichever company that manages to hit the Amiga pinata and win the candy that real work on the OS must be done, and done quickly. Because it isn't the developers that stand in the Amiga's way, it's the Amiga. It's the proprietary technology that once held its head high that causes it to swing low; it's the Amiga's undeveloped OS that now trails in its ability to provide developers with a simple, effective, and work-reducing environment from which to build the large applications that today's information market wants to be sold -- something that, you will note, NeXTStep is most often heralded for. The tools developers have on that platform is unmatched, and I'm sure even Microsoft will be playing catch-up for years.

With the lack of development tools, the poor documentation provided with the OS, the death of AmigaMail, the absence of the AmigaWorld Technical Journal, and even that of AmigaWorld itself (not that it was ever truly informative), I dub this the Amiga Report Technical Journal, hoping to fill all of the above voids. I'll do my best to provide documentation on the new OS functions that have been so misused or underused; pen allocation, BOOPSI, DataTypes, and more. Not all at once, mind you, or I'll run out of readers; and I won't hit you with a big brick, because these aren't necessarily easy, or obvious concepts to swallow; as well, I hope to grant you all with the blessing that is my wisdom, vision, and intellect, with a side order of cynical comments. Because there's more to the Amiga, and the Amiga's OS, then just AutoDocs and example code;

there's vision. And that vision has been lost by many, and it is the one thing the Amiga needs most to survive; without the vision, there would have been no Amiga in the first place.

Without vision, there will be none in the future.

Sincerely,

Gregory R. Block, Editor of the Amiga Report Technical Journal

1.4 Liftoff!, by Jason Compton

Liftoff!

By Jason Compton, Supervising Editor

It's been a real experience writing for, and more recently writing for AND editing Amiga Report Magazine. It's been a chance to take a good idea and turn it into an (if I may say so) amazingly successful and popular source of information. Of course, the argument can always be made that the Commodore liquidation proceedings have created a huge void of news that we just happened to fit neatly into, but I'd like to think that it rested on the hard work of the writers and the support of the readers.

A few months ago, I started thinking of what AR was missing and what job it could or should be doing. Then I started thinking about what the Amiga was missing...and it came to me: an online technical journal, bringing the sort of rapid distribution the format allows to a new audience, with the sort of information that might not QUITE fit into the much more broadly-based Amiga Report. To put in development material, in-depth tutorials and articles, and...well, whatever else happens to come our way. :)

That's where Greg steps in. I knew AR Tech needed an editor other than myself...first off, I simply don't have the sort of expertise it takes, and even if I did, I wouldn't have the time to do the job right. Greg's already set a pretty high standard and set of goals for the magazine, which thrills me. (After all, it'll be an online magazine I can READ for a change!)

Please enjoy yourselves. The Amiga needs people who know how the machine works, particularly with the liquidation's conclusion just around the corner. The auction is April 20th...just three weeks away as I write this. The long sleep may be coming to an end.

Jason

1.5 Programming in Zen

Programming in Zen

You can give the water in an empty well to a thousand armies, but dividing the butterfly by nothing will leave you with butterfly intestines and a

mess to clean.

If YOU have a funny programmer's mistake you can make into a zen-like rule of thumb for coding, email it to Zen@Oubliette.com

1.6 Letters to the Editor

From: Winter@Oubliette.com (Gregory R. Block)
To: Editor@Oubliette.com

Dude, you suck. Totally.

Yeah, buddy, well, you're a schizo. Zark off.

If you'd like to write to the editor, and maybe, if he feels good, get stuck in this column, write to Editor@Oubliette.com.

1.7 Pointers -- Hot spots, news, and files

```
#include "pointerdefs.h"
/* Pointers.c */

ULONG Main(int Attraction)
{
    Object *Pointers = NULL;

    Pointers=NewObject(PointerClass, NULL,
        POINTER_FTP,    NewObject( FTPPointerClass, NULL,
            FTP_Site,        AMINET,
            FTP_Filename,    "gfx/board/EGSA2410v1.0.lha",
            FTP_ShortDesc,   "The latest A2410 beta EGS driver.",
            FTP_FullDesc,    "gfx/board/EGSA2410v1.0.readme",
            FTP_Flags,       FTP_BETA | FTP_COOL | FTP_PROMISING,
            TAG_DONE),

        POINTER_FTP,    NewObject( FTPPointerClass, NULL,
            FTP_Site,        AMINET,
            FTP_Filename,    "dev/e/epd15.lha",
            FTP_ShortDesc,   "Serial publication on Amiga E development",
            FTP_Flags,       FTP_USEFUL | FTP_DEVELOPMENT | FTP_E,
            TAG_DONE),

        POINTER_FTP,    NewObject( FTPPointerClass, NULL,
            FTP_Site,        AMINET,
            FTP_Filename,    "util/dtype/ZGIFDT39.13.lha",
            FTP_ShortDesc,   "ZGif datatype update, fastest one around.",
            FTP_Flags,       FTP_OS3x | FTP_DATATYPE | FTP_MUSTHAVE,
            TAG_DONE),

        POINTER_FTP,    NewObject( FTPPointerClass, NULL,
```

```

        FTP_Site,          AMINET,
        FTP_Filename,      "util/dtype/cdt_39.6.lha",
        FTP_ShortDesc,     "Totally cool C source code datatype.",
        FTP_Flags,         FTP_OS3x | FTP_DATATYPE | FTP_MUSTHAVE,
        TAG_DONE),

    POINTER_WWW,          NewObject( WWWPointerClass, NULL,
        WWW_URL,           <A HREF="http://www.whim.com/~guardian">"http ↵
                           ://www.whim.com/~guardian/"</A>,
        WWW_ShortDesc,     "Andrew 'Guardian' Denton's Home Page",
        WWW_Flag,          WWW_COOL | WWW_GRAPHICSINTENSIVE | WWW_HUGE,
        TAG_DONE),

    POINTER_Sexy,         attraction,
    TAG_DONE);
if (Pointers)
{
    DisposeObject(Pointers)
    return ((FUNKY | COOL | THRIVING));
}
else return(NULL);
}

/*
 * If you would like to see your pointer entered into next month's Main( ↵
 *   Attraction),
 *   EMail your struct definition to Pointers@Oubliette.com.
 */

```

1.8 The Man from M.Y.S.T.I.K.

An interview with Dave Reed, creator, designer, and mastermind behind the opus in progress known only as "Mystik Tank".

ARTech	Okay. Let's start this off at the beginning, the obvious falsity, and work our way up. Is texture mapping impossible on the Amiga?
davereed	Impossible is a funny word. Such techniques are possible on any machine with enough memory to support textures.
ARTech	Okay, so what does the Amiga architecture present as difficulties when trying to build a texture mapping algorithm?
davereed	As many people mention, it does take more work to do this kind of pixel by pixel handling with a Planar chip set; though, with faster machines, texture mapping still works well. It's often a matter of money.
davereed	People in the PC market often will buy the highest and most powerful machine they can get their hands on for a game.
ARTech	So, what kind of Amiga can honestly be expected to do texture mapping well?

davereed Any machine with the Mhz to push it. If the standard was the 040 ←
processor, texture mapping would
hardly be an issue.

ARTech So a good texture mapping algorithm on the Amiga is 100%
reliant on the speed of the main CPU, then?

davereed Done the easy way, yes. Done the smart way, no.

ARTech So, what kind of algorithms are being used, by and large?
Are we still following in the footsteps of DOOM, or have the
algorithms become more complex?

davereed To tell the truth, some algorithms have gone past Doom, at
a price of speed, no less.

ARTech What kinds of algorithms are the easy way, and which ones
fall under the "smart" category? Is there still a use for the
Amiga chipset in the advanced algorithms that texture mapping
presents?

davereed Well, hard for me to say. I don't handle the technical aspect of these ←
things too much.

davereed You can actually get some nice results by using some of the
cycles in the Custom Chips, but lots of people worry about
making something too custom chip dependant.

ARTech Do you?

davereed Basically, I handle the game design and ideas behind it, but
the coders do most of the techie stuff.

ARTech I see. How many coders do you have working on your engine,
then? And how many man hours have been spent in the design
phase?

davereed Okay, one question at a time.

 (ARTech smiles.)

davereed Right now, there are 2 coders working on the main game engine.
The main engine is a 2 part series.

davereed The design phase has been long. too long. I mean, too long,
for it is very hard to get help when you aren't well known.

ARTech What kind of help have you needed?

davereed It took me 2 1/2 years just to get a team that has stuck together.

davereed I used to try joining people who started groups, but they
often fell apart.

 (room fades in one of those wavy wipes into a dream sequence)

davereed About 5 years ago, I bagen writing fantasy stories. They were

not simply fantasy stories, but they were somewhat of an autobiography of myself.

davereed I wrote them, initially, for self-illumination. Something to help me (perhaps) grow wiser in the ways of life (so to speak). I just never liked to write about myself.

davereed After a couple of years and Wolfenstein 3D was released (also, after 3 stories I wrote, based on the same planet), I had an Idea for a game.

ARTech Idea with a capitol I, it seems.

(ARTech smiles.)

davereed Yes, a capital I.

davereed A futuristic game to take place many years, after these stories had taken place. Now, I can really say that this may be the first game designed with 5 years of storywork behind it that hasn't already been made a series, or a movie, or something.

davereed Well, Wolf 3d awakened the memories of the fun Atari 3D arcades i used to play. I've played nearly all the arcades, ever since the Pacman era.

ARTech So did I. Long gone are the days of typing in programs from Compute!, though.

davereed Heh. I remember typing in programs from Compute!

ARTech I've found that there are few programmers that don't remember typing in programs from Compute!

(ARTech smiles.)

davereed I started to design my dream game.

ARTech Tell me about the dream game. (no, not you, Usul.)

davereed Well, I started on developing the idea of a game that mixed the 3d Atari Arcades (With things coming at you from all around), the environment from the movie Tron, and the kind of Magic system from an RPG.

ARTech Sounds like quite the dream.

(the earth shakes and splits as IRC undergoes one of the hundreds of netsplits that rock its foundation daily)

davereed The Old arcades also didn't mind doing a split screen idea, with more than 1 perspective.

davereed Often a double perspective, for 2 people to hack around.

davereed What the double display is used for in my game, I'm not sure

I can say, at the moment.

davereed But lets just say that Tanks are often not controlled by 1 person.

ARTech Okay. So, we come back to the design, then; have your programmers asked you to scale down your vision of the game?

davereed One has, the other is doing another game which also does some pushing of its own.

davereed Me. I don't ask to scale things down. We take things one thing at a time and just keep adding, till it reaches what it is supposed to do. Then we talk scaling.

davereed To make an ok game, you code simply thinking that we have to cut this and this out.

davereed To make a game that truly is innovative, you've got to simply code for what you'd like to have, and then once the game is there, see what you can do to it.

ARTech Do you think that the problems you had trying to get programmers to work with you are in any way symbolic of the general attitudes of the Amiga development community?

davereed Some of them, were mainly problems between themselves.

davereed The idea that the Amiga has some effect in the difficulty of finding anybody is true.

davereed That's why Origin took off so easily.

davereed The other people weren't working for me, but I was coming in as a musician, trying to break some known ground. To establish myself so that I could be in a position to make the game a reality with that group.

davereed Scratch Musician. Composer is the better name.

ARTech I've had people say that while the users seemed to have a strong vision of the Amiga, somewhere along the line, the programmers lose it. Do you feel that has any truth? Is something disillusioning our development community causing them to lose the vigor they held as users?

davereed Fear is the answer to your question.

ARTech Very Kosh-ish of you. So, you think that it's the times we live in, then, and not a general attitude.

davereed Origin, in the PC market coded for the biggest machines with games that were spectacular enough to get people to upgrade.

davereed Many programmers in the Amiga market are afraid to do what Origin did because they aren't sure of Amiga users will upgrade for a new and spectacular game, such as in the PC

market.

davereed Years ago, I believed Amiga users would. I still do.

davereed Not that the Ami scene is the only thing I plan to code for. But to push the market for faster processors, I'm all for it.

ARTech What kind of requirements, then, can users expect your dream game to have? 030? 040? Graphics card? ECS or AGA?

davereed As for the first project, mainly what is needed is 2 megs and for the speed a nice 030. Graphics is not dependant.

ARTech And the second?

davereed The second should be the same, based on the same engine.

davereed I do have something in the platform genre working up, on paper. I've even started some of the soundtrack for that one, but that would be AGA.

ARTech What kinds of problems have you had designing an engine that is both depth and hardware independent? What kinds of things have you done to make sure that the hardware is used to its potential?

davereed Well, with chunky to planar conversion, there isn't an AGA hardware specialty that is needed. But Magical spell effects are nice with custom chips. There is also some other things such as transparencies, etc....

davereed Take Black crypt for instance.

ARTech Please do. I never had the opportunity to play the game.

davereed Black Crypt really doesn't need a whole lot in the custom chips, but the magical effects use the blitter nicely.

davereed When you play around with what you can do with a plane in planar graphics, they can do some fun things without the CPU.

ARTech What kinds of things can be done with the planar graphics system that you feel are the best examples of that statement?

davereed One of the things I like about Amiga's custom chips, in particular, is the transparent copper runs.

davereed It is done in some euro demos, but also in Black Crypt (Yep, there's that word again).

davereed In BC, there are various magical barriers which you can see through, but have their own colors which cycle in a particular direction. (Half transparent walls).

davereed They don't have to be walls, however. Imagine, using something that cycles in a direction to refract or reflect light.

davereed Just an idea.

ARTech Fascinating.

davereed Well, BC doesn't shoot off light. I was just taking it to a possible conclusion, could make a neat idea of something to bounce things off (i.e. lasers possibly?)

davereed We are coding the stuff as system friendly as possible. I.E. no real poking around. There are some things in which the graphics are real easily ported between the two sides.

davereed For the first versions, the game will need the custom chips.

ARTech What happens when the custom chips are gone? Do you then rely on the CPU to do what the chips would have done, or do you simply leave the feature out?

davereed Under RTG, most machines will be powerful enough to do the same things without them.

ARTech So you plan on releasing a version of the game for each "platform" then; such as ECS, AGA, and RTG-boards?

davereed The engine already supports ECS and AGA. RTG boards are an option, only needing to redirect some thing.

ARTech How have your programmers reacted to the demands your dream game has placed upon them? How do they feel about the project?

ARTech (one would hope that they are ecstatic, and honored, but you know how programmers can be.)

davereed They keep telling me about how big and well-done the storyline is. They see a neat idea, but I know they aren't sure about how it is until they have a tangible product ready.

davereed Perhaps they make think I'm a bit looney, especially that I wanted a combination texture mapper and polygon system put into the engine. I.e. a double engine.

ARTech Sounds like a good idea to me, though.

davereed It'll work. Years after I came up with that idea, Team 17 is now working on 3D Off Road, which does this.

davereed Imagine, if I could've gotten the team as early as I wanted. I would've had the game ready before Doom was ever released to the public eye. :)

davereed The game is nothing like Doom, in game play, but people will see texture-mapping and immediately classify it.

ARTech I know that you and your team work over the internet; how has that helped or hindered productivity, and what has it done to the communication between design and coding phases?

davereed The design definitely gets hindered by internet, but mostly by not having a mailing list.

davereed Things seem to be moving along, however, mainly since I come on here and mail so often.

ARTech So you feel that for proper development, the minimum requirement for productivity would be a mailing list.

davereed I basically try to maintain a daily presence, here. If anything comes through, I relay it.

davereed I also try to ask questions and so forth,

ARTech Have you found productivity within the constrained walls of IRC?

davereed Funny enough. Dave Bryson and I are the only ones with IRC access. At least, the only ones I've seen on IRC.

ARTech Has it been useful?

davereed IRC. Hmm. It's been useful for my general Amiga stuff, but not much for the game.

ARTech Mostly email, then.

davereed Heh. my project has been all email and/or ftp.

davereed The design of the game is big. The tmapping only makes a tiny part of the game.

davereed I had originally designed the game without any texture mapping, but Tomwoof insisted that I do it.

ARTech How many people, total, are involved in creating various elements of the game?

davereed I can say around 5, at the moment. Two for the engine, two others for the Cartoon stuff.

davereed There are other people doing voices and stuff, but the voices come last in the animation process. That, is the fun part!!

davereed Basically, the people speak into a microphone as they watch the character's mouth move.

ARTech Wow. :)

ARTech Is there anything you'd like to say to anyone who, like you, wakes one morning with one of those Ideas for a game?

davereed Heh. No one can wake up one morning and make a game.

ARTech Yes, but is that what you want to SAY to them. :)

davereed Nah!

ARTech What would you say, then?

davereed All I can say is: If you have an idea that you REALLY want to express. Write it down and don't ever, ever, *EVER* give up.

davereed Did I mention to never give up??

davereed I can't say that more than enough.

davereed I'm not pushing this game to simply make a neat game. I have to finish this. There's a huge message behind it and all of the stories I will probably have published when the game is released.

ARTech Thanks, Dave. Now's the part where you give us your contact information, so that anyone who thinks you're a god can write and worship you.

davereed Heh. Worship is the last thing I want.

ARTech I'm sure all of the parents whose nobility many computer games seem to offend will be appeased by that statement alone. :)

davereed That's the funny part about my game. It couldn't be considered either super violent or super cute.

davereed And I never like to be seen as higher than anybody else. But my main email address is davereed@wam.umd.edu

ARTech davereed: Do you have a working or final title for the name of your game?

davereed Yes, the name is Mystik Tank (Spelled correctly for the title)

davereed It's a funny name, but it fits the description, once you know the storyline. It's also not simply about the Tank. :)

ARTech Thanks, dave. And, of course, you're free to close with any final comments.

davereed Ok. I actually do have a comment.

davereed Basically, it is for developers, no matter what platform they are on.

davereed when you come up with an idea, think first about what it can do, not how small you can make it. Tone it down later, not before. Otherwise, you never know what you could've missed.

davereed The programmers who do this will always come out on top.

davereed And the funny thing, they'd think it was like taking candy from a baby. Not to condone the stealing from babies.

(davereed smiles.)

ARTech Sounds like good advice. Not the candy from babies, of course. :)

(ARTech smiles.)

ARTech Thanks for your large chunk of time. :)

davereed No problem.

Dave Reed can be reached at davereed@wam.umd.edu.

If you'd like to be interviewed, or know someone who would or should be, send email ARTech@Oubliette.com.

1.9 Conference: The Amiga-OS Project

The AmigaOS conference, held in IRC channel #Amiga-OS on 4/4/95

The AmigaOS conference, held on what is becoming a very popular channel on IRC, is one of the examples of developers taking things into their own hands; if nobody does the work, everything stagnates; these people, whether participating in discussion or writing code, all deserve more credit than they get. Because no matter how difficult writing code is, writing good code is one leap harder; and no matter how good that code is, without an interface and a plan of implementation, it's meaningless. Here's to the planners, the thinkers, and the shakers out there.

Tau suggested vote topics: binary compat, memprot, resource tracking, multiuser filesystem, clone 3.1/go for new

stefanb Tau: Binary compatibility = NO, all others seem OK.

Tau sb, why not? can load old binaries, doesn't mean every new feature oughta work with them

mlelstv tau, why new binaries ?

Tau mlel, stefan just said no to binary compat.

stefanb Tau: We should fix all brain-dead system calls. OK, if we make "compat" libraries for old binaries. The new libraries should have newly designed System calls.

mlelstv tau, can't understand that..

BM stefanb, Agreed. :)

mlelstv tau, I fear we get a committee decision..

Tau stefan, I'm against that.. we should keep it as compatible as

possible, but not be afraid of breaking some things if an essential feature demands it

stefanb ml: Maybe there is no "correct" answer. Then we have to make a decision what _most_ programmers want.

Lynet Tau: How do we add MP and RT without breaking 99% of the old apps?

stefanb Tau: Most essential new features require breaking the old stuff.

mlelstv stef, as I said.. then either some people will leave the project or we have two teams working on different designs.

griesbrei Lynet: RT with an extra library

Lynet griesbrei: Use cooperative RT?

mlelstv stef, as for me. I am not interested in a 'I can do it better' -design.

stefanb ml: Then this only means that Amiga people are the same as UNIX people. They can't work together for a common goal.

griesbrei lynet: cooperative??

Tau lynet, I posted a long article about how I'd do it on the mailing list some time back. Seems to me device drivers would break, most other things could be left functional

Tau stefan, UNIX people can work together?!

stefanb Tau: They can't. example: NetBSF <-> FreeBSD.

Lynet griesbrei: I.e. that each application has to tell the OS it wants RT.

Pepo resource tracking comes free with an OO system. Old programs would simply not be tracked.

griesbrei maybe the first thing we should add is a global Amiga time ;)

WilloW No I'm not kidding.. he didn't receive the announce message you posted.. Really wierd.. he's quite busy also..

Arcade Ah ... that's bad.

stefanb Pepo: We could do "compatibility" libraries for old binaries.

Arcade Indeed ... that's what I propose.

stefanb Pepo: They would be mapped to the new calls.

Tau gries, pisses me off I have to actually adjust the clock for daylight savings.. Locale should remember that for me, just like it does for my timezone ;)

griesbrei lynet: that's it. add a hook for each resource you want to be freed

BM Stub like stuff yo u mean?

Pepo stefanb: oh, you can't build an whitebox api over an blackbox api. the new stuff has to be based on the old stuff, not the other way

around.

Arcade tau: that's a thing to add ... :)

griesbrei tau: maybe some sort of rsadio controlled clock..mmmh

mlelstv pepo, nah. everything gets translated for and back :)

BM Nah, Synclock is fine for that. :)

PieMan hates daylight savings

Arcade Or syncro.

WilloW yes!!!!

Pepo mlelstv: it's impossible

Tau I actually had to adjust my time zone to russian time (-3) to keep my clock in time.. since it's adjusted automatically from my host

mlelstv pepo, you can simulate everything.. look at multifinder :)

griesbrei mlel: why not simulate a Cray ;)

stefanb Pepo: Then compatibility is completely impossible, even if we leave the old calls in and add new ones.

Tau mlel, umm.. that makes my eyes sore ;)

Murty Okay, I'm here. Sorry I'm so late, guys : to be honest, I just forgot the entire conference ! :-(

Arcade Okay guys, welcome to this channel. Sorry if it means an inconvenience to some of you.

stefanb gries: Who wants such a piece of crappy hardware?

Pepo stefanb: huh? how does leaving old stuff in break compatibility?

mlelstv tau, well, that's why I don't like the 'complete new OS that simulates the old'-approach.

griesbrei stefan: yep, i like my pocket calculator most..

stefanb Pepo: You still have your white-box API in. So the new black-box API can't work securely.

mlelstv tau, in fact, if we want to do that we should concentrate on an existing kernel such as Mach..

Tau mlel, me neither

griesbrei SO, CAN WE START?

Pepo stefanb: we are not UNIX. We don't need a secure OS.

Arcade Why would we need to use a kernel such as Mach?

stefanb ml: Not everything has to go, only the wrong decisions have to be removed.

mlelstv tau, problem with that is you need big machines to get something reasonably fast.

stefanb Pepo: secure = stable.

Tau exactly.. if we were to build emulation libraries, we might as well write them for some existing OS
mlelstv stef, wrong decisions such as FindTask() and Disable() ?
BM Security is essential for multi user machines, not single user ones.

stefanb ml: What's wrong with FindTask()?

mlelstv stefanb, direct access to the TCB

BM Forbid()?

Arcade How about networking? multiuser security is valuable then.

Pepo stefanb: _old_ programs wouldn't benefit from the OO "overlay", but new ones will

griesbrei stefan: it cannot return easter eggs..

stefanb Pepo: But the OO overlay can't be stable if you have an old app running, which uses white-box access.

Murty Arcade : when you do networking, multiuser security is a *must*.

BM Hmm, not at the process level though, checking ownership of everything would be a drag.

Tau has work on dos started, btw?

Murty BM : No, I'm talking file system level here.

Pepo stefanb: how would it work with emulation libraries then?

mlelstv stef, without memory protection nothing is stable..

griesbrei tau: in fact, we didn't start on anything yet :(

Tau murty, OS 3.0 already supports multiuser at filesystem level.

BM Oh, for sure, some superficial security would be neat to have.

stefanb ml: VMem is easy, just allow Pools with VMem...

Murty Tau : yes, but don't ask how. There are security gaps in 3.0 and even in MultiUser.

mlelstv stefanb, I do know that.. I was talking about protecting memory.

Arcade tau: the problem is that program's dont. having basic support in the OS makes programs support it too.

stefanb ml: I was just pointing out to Pepo that if his argument is right, then we CAN'T produce a compatible OS.

Pepo stefanb: so you want to _interpret_ the programs to keep track of what fields in a structure are written to?

mlelstv stefanb, why not ?

Pepo stefanb: old programs, I mean?

Tau arcade, what basic support?

stefanb ml: Compatibility Libraries or new system calls in the old libraries are virtually the same.

mlelstv stef, my idea is to have pools mapped to memory regions (lists of MMU pages). A task can have private pools. The loader could allocate private pools for code segments, etc..

Arcade pepo: we would wan't to have a basic multiuser security, not very severe. Something like muFS, but better support from within the OS.

BM Arcade, Agreed.

Murty stefanb: When was the last time you received anything ?

Tau arcade, give an example, how is filesystem level support on OS 3.0/MuFS not enough?

BM Can you store ownship info in 3.0?

Lynet Arcade: AmigaOS is a single user design. Why clutter it with multiuser support?

Tau you can't Open() a file you have no access privilege to.

That's what the OS is supposed to make sure

Pepo stefanb: just how would you make old intuition gadgets work, when the "native" OS only supported gadget objects (boopsi or gadtools)?

Murty stefanb: Well, at least it seems there isn't a problem with your address and the list then.

Arcade tau: It doesn't provide for other things like disabling the bootmenu, formatting harddrives, etc.

Beefeater Lynet: When you hook it up in a network

mlelstv bm, yes. you have 2 words for user id and group id. isn't used by the filesystem for access control though.

Tau bm, FastFileSystem doesn't store ownership, but muFS/Envoy do. dos V39 has the calls necessary

stefanb all: Shouldn't we start with a first topic and then discuss the next?

mlelstv tau, fastfilesystem stores ownership!

Lynet Beefeater: When you hook your Amiga to a network, let the networkservice daemons take care of multiuser security.

Murty stefanb: Yes, I agree. Let's take it one thing at a time.

Arcade And access control is what we need (or want). I can be disabled though.

Tau mlel, well, it doesn't use it then.

mlelstv tau, it's just ignored. Envoy evaluates the uid/gid

Arcade Okay, let's talk about binary compatibility for now ...

stefanb Murty: Go ahead, set the first discussion point. I think the

main talk should be about Exec first...

Murty Binary compatibility it be. There are two things to be considered here...

mlelstv arcade, we do need binary compatibility. the OS should run on existing machines with existing programs.

Murty ...one : we can easily use older programs...

Arcade Would it be a bad idea to just write a 'new' OS, and make it possible for old apps to use our 'emulation' libraries like intuition.library etc.?

Murty ...two : we are restricted in adding new features and/or changing fundamental things when we do go for binary compatibility.

Tau arcade, yes. why write the OS in the first place, then? Why not just make an Amiga emulator for OS/2, Hurd, Linux or something?

BM If it can boot on an amiga you can choose to boot it or not, no?

Murty Well, we can't make the OS OO-oriented from the bottom up, when we wish to go for binary comp., because...

Arcade Because we want to take ourselves and others to the future?

rozga KMEL: Please drop me a mail after this disc.

(rossi@tindrum.tng.oche.de)

Murty ...then library calls would fail. Unless we make a lot of emulation libraries, which would mean emulating the OS and not so much rewriting it.

Murty (as you can see, I'm writing in pieces :)

griesbrei I guess we all want to use existing programs, don't we?

Murty Emulating the OS is bound to be slow for older programs, which is all we'll have at first.

mlelstv murty, why would library calls fail ? I do not want to junk all software that exists.

Murty griesbrei : I know I do. If I wanted to switch to new software, I would buy another computer.

Pepo griesbrei: as long as they are coded to the rulez: yes.

Arcade True. One would be able to boot the 'old' OS anyway. And these programs will certainly not work on other hardware anyway.

Murty mlelstv: if library calls are redesigned, they will respond differently and programs who rely on the old method will break.

Arcade Every program that is system compliant, would run on these libraries.

PieMan I'm afraid the hard truth is everyone is going to want to use

old software that would be incompatible with our OS.. as a result everyone says "hey WB 3.1 ain't so bad" and our project is a flop

Pepo Arcade: SWITCH? I.e. not being able to run them TOGETHER? :-(

Kudo I think binary compatibility will make it harder to make a good OS with new features

mlelstv murty, no. the library calls define the interface to programs.

these library calls must exist. otherwise it is not AmigaOS

anymore but something completely new.

Arcade Pepo: Not concurrently, no. Impossible.

Tau arcade, I won't be rebooting my machine every time I want to use another program

griesbrei PieMan: Nobody will use a new OS without any software..

BM mlel, To a point. If the calls are only slightly different then porting is a snap.

Murty mlelstv: Exactly. If you change the design so fundamentally that the library calls change...

Arcade pepo: But if we do our job properly, one would be able to run old programs under the new OS. System-compliant programs, that is.

Murty ...i.e. no binary compatibility...

mlelstv murty, then this is no longer AmigaOS. I don't think that anybody wants that.

griesbrei murty: so we need new libraries!

Lynet mlelstv: But adding memory protection and resource tracking without changing some of the API would be very difficult.

Murty ...then you will have a different API and you will indeed have to run 3.1 to run old software.

PieMan griesbreg: i agree.. and they have a choice so they just won't choose it...

mlelstv bm, the calls have to be the same, to the word of the documentation

Kudo think it would be better to go for source compatibility

Arcade Yes. We would use new libraries instead of the old ones, the old ones would be used for compatibility.

Pepo Arcade: I remember reading something different, but this is the way to go, old programs _should_ work

Murty mlelstv: I agree with you ! I do ! I want to keep using my old software, and making the library calls compatible is the only

way to do that.

Tau If the calls are only slightly different, WHY change them at all?

mlelstv lynet, indeed. with (full) memory protection, you have to write a complete new and incompatible OS

BM mlel, existing documentation of 3.x? Or New docs?

Arcade Because the underlying system would be very different.

Kudo murty: if we only want to use old software, There's no use writing a new OS.

mlelstv bm, existing documentation of 3.x.

Murty However, if we would go 'for the future' with a completely new OS with MP and all, it would be very hard to keep the old library calls.

mlelstv bm, I insist in compatibility to the word. we do not have to copy the bugs :)

Lynet mlelstv: I understand that. The AmigaOS is so dependant on publically available structures that adding full MP would be impossible.

Murty Kudo: Yes there is. Say your Amiga dies or becomes too slow. Then you can get a new machine and run your old software on it.

mlelstv murty, it would very hard to keep existing software..

Kudo murty: but not the same binarys...

BM mlel, I'd like a cleanup and simplify done to 3.x. :)

Pepo OK, let me make this point clear: If the ultimate goal is to write a new OS, that doesn't look and feel like AmigaOS (from the _programmers_ standpoint of view), I'll leave the project.

Arcade Murty: Only if these programs would be recompiled anyway.

Kudo murty: unless it's a 68k machine

Murty mlelstv: No, I don't think so. You'd have to recompile the source for a new processor, but that's about it.

griesbrei We needn't keep the old software until the end of the world!

mlelstv lynet, yes.. and no. the number of public structures isn't that large. it mainly affects hacks and small utils

stefanb Pepo: That's NOT the goal.

BM Look and feel no problem, but strict compliance would be redundant.

mlelstv stefanb, sure about that ? I am not.

stefanb Pepo: But IMHO we have to make the decision if we want to fix the design failures in some of the current calls.

Pepo stefanb: actually, memory protection _would_ make the OS look completely different.

Arcade pepo: We would want a very similar system, as similar as possible but with new features and portability.

ZinO 'll be back in an half an hour..

Murty Pepo: MP makes it almost impossible to stay compatible.

Kudo agrees with Arcade

stefanb Pepo: And the question is, is this possible with adding just some replacements to the current libraries which are ONLY used by new programs.

desrat if we do have MP..please make it optional

griesbrei Can't we put old stucts in public mem?

mlelstv murty, mp as a security feature: yes, as a debugging aid and crash protection: no.

Arcade We have to. Not everybody has an MMU (like me). :(

Tau MP should be a user option, just like it is on OS/2. For different reasons, of course...

Kudo Think we should stay as (source) compatible as possible, but not if it will sacrifice efficency or something else

BM Optional MP would be required if we want anyone to actually try it out. :)

mlelstv griesbrei, yes. but if these are not protected you can crash the system.

Murty Tau: if we put MP in, we will have to make a very big effort to squeeze it in and be able to use old software at all. Making it optional too will only make things even more complicated.

Pepo stefanb: "this" ?

BM Really secure memory protection would be almost impossible I think.

Arcade You can't protect public accessible structures, it will break everything. At least under the current OS.

stefanb Pepo: Replacint brain-dead functions.

Tau bm, partial's better than nothing

Pepo stefanb: replace as in remove?

Murty Arcade: Yes, you will break everything. I say forget the MP and go for as much compatibility as possible.

stefanb Murty: MP for Pools is the best option for this.

Tau arcade you don't protect on a structure by structure level anyway

stefanb Murty: BUT the OS has to know about it and use MP pools as far

as possible.

Arcade No, that would be impossible with the MMU pages.

Murty stefanb: How hard are pools to implement in the current API ? I wouldn't know, really.

Arcade Pools are already there, I believe.

Murty I mean MP pools, of course.

mlelstv murty, I think it is quite easy to get protected and virtual pools.

stefanb Pepo: THAT is the question: remove -> compatibility libraries,
replace -> add them to the existing libraries.

Pepo every new app could call the exec function ProtectMe() in
startup and all allocmem's (non-public) would be protected, and
even automatically put in a pool which is freed at RemTask()

Murty mlelstv: This would be a good way to implement MP, make it
optional and (thus) be able to remain compatible by switching it
off.

Murty But can it be done with MP pools ?

BM I think flags during memory calls would be ideal.

stefanb Murty: Just add an option for MP for the already existing memory
pools.

mlelstv pepo, could even be a feature of the loader.

Pepo mlelstv: agreed, magic hunktypes :)

zchu1121 pepo: You could as well use the uid.

Murty stefanb: This would mean that only newer software can use MP,
right ?

mlelstv pepo, you even have a supported extension of hunk memory type
that isn't utilized yet.

griesbrei new software uses the new funktions and may use MP

mlelstv murty, the user would never know whether memory is protected or
not. Every call will work with or without protection.

griesbrei old software uses the old functions and gets ONLY public

griesbrei so what's the problem?????

Murty griesbrei: If the new software is the only one to use the MP,
then it would go to the list 'additional features' in my book.

stefanb BM: Not needed, if a pool is MP then all memory in the pool is.

Arcade We could actually have something like that in the system, having
a program ask for its memory to be protected. But it would not
be very secure.

mlelstv bm, might be not enough.. but that's a detail..

Murty And new features are lower priority than getting an OS up and
running, IMHO. They can be added later.

stefanb BM: If the hardware doesn't support MP, then the MP flag for pools will be ignored.

Pepo stefanb: if you do not want to bloat the OS, it makes sense to extend the existing libraries, and add new calls. you can do much code reusing internally this way.

griesbrei murty: what is an additional feature? MP?

mlelstv stef, you need new calls to handle ownership of pools.

stefanb Murty: Of course, because you have to design it for MP.

Murty griesbrei: Yes, if only used by new programs, then it would be a New Feature (tm).

mlelstv stef, I'd also make 'sets of pools' to avoid using half-full memory pages.

desrat murty:i agree....we have a greater need to get the new os up and running

BM Hmm, I think some allocations would have to be public, hence flags.

mlelstv stef, yes. I have a clear view of what such pools should do :)

Murty Okay, so suppose we do MP by pools, then we have to put that in the Exec design. Is Chris Gordon here, by any chance ?

griesbrei murty: we CANNOT add MP for old software.

stefanb But IMHO the biggest problem is that the OS wasn't designed to use MP memory...

Murty griesbrei: No, that's what I said.

Arcade Anybody with a real good reason for not making a new OS, and provide compatibility libraries *speak now*.

Arcade (we should get on to the next subject, on which we already were)

stefanb But I see no problems e.g. for Intuition to use MP, because onbly the program and intuition are _normally_ allowed to access the user interface strucutres for the program.

Murty Arcade: I just said we should be weary of the compatibility libraries, because we don't want to write an Amiga *emulator* !

griesbrei Murty: we NEED an emulator or no one will use our OS (no software)

Murty griesbrei: No, we need an OS that runs the applications in native mode. That will run old applications, but it won't have to translate all the old calls to new ones, like an emulator would have to.

Pepo stefanb: think what OpenWindow() does... it needs to change the NextWindow field of another tasks window...

griesbrei so, we'll write a completely new OS and leave the old libraries. That`s it?

mlelstv arcade, a new OS is very difficult to test.. a rewrite and enhancement of the OS will allow partial testing

stefanb Pepo: But Intuition is SYSTEM SW, it is allowed to change SYSTEM structures.

Murty griesbrei: We'll extend the functionality of the old libraries, retaining their original structure and API.

Arcade mlelstv: How would you do that, SetFunction() every function in a library?

Pepo stefanb: but how would you protect another task doing it?

stefanb gries: And only NEW programs will be able to use the NEW calls.

griesbrei murty: and what if there comes an AmigaOS 4.0?

Pepo stefanb: run intuition in supervisor mode? change MMU tables? ugh...

stefanb Pepo: Decalre it an illegal operation. we are talking about running SYSTEM conformant programs.

mlelstv pepo, a window structure belongs to intuition. it might grant individual tasks read access.

Murty griesbrei: I've sent mail to CBM UK and I heard David Pleasance was going to contact me, but I've not heard anything yet.

Pepo stefanb: the point is you want it to be _secure by adding pritection_ instead of _secure by relying on clean software_

mlelstv pepo, problem is that intuition would need a pool for each client :-/

Murty If there will be an AmigaOS 4, we would have to cooperate with its authors. No use in having two rival OSES.

rozga Isnt the point wether we want binary compatibility first or new features first?

stefanb ml: The pool would belong to the client. Because he is already using MP pools.

Arcade rozga: exactly, that's mainly the intention of vote #1.

Murty griesbrei: No, but if we have done a lot of work, they may want to benefit from it. And we may want something nice in return :-)

mlelstv stefanb, but intuition allocates it... from a clients pool ???

stefanb ml: Yep.

griesbrei Could you stop that thing on MP for a while please??

rozga Arcade: Did I miss the Vote #1?

desrat murty: a nice 4000 per contributor would be nice...)

Arcade PieMan: whether we want binary compatibility or not.

BM I think features first but that the features will demand changes to binaries.

Tau m1el, why ugly? wouldn't the call be done on client's context?

Murty desrat: :-)) I was thinking more along the lines of being kept up to date on OS developments and a free registered developer membership for everyone, or something.

griesbrei Murty: Don't want to work for C=, I want a complete PUBLIC os.

stefanb OK, vote #1: compatible OS, existing API. New features are only available through NEW API calls.

m1elstv tau, unsure.. intuition has to access its own structures too. it should not do this on the callers context

Pepo Tau: right, allocations of intuition objects should only be done on the context of the apps task

stefanb m1. Intuitions own structures are allocated on Intuitions task pool.

Murty Yes, Stefan. How will we do the vote ? Everybody be quiet for a while and two raises of hands in the shape of PINGs ?

m1elstv stef, I'd add new features to old calls too. like new tags or flags.

m1elstv stefanb, a call to intuition has to access multiple pools. how do you handle that if the pools are protected ?

stefanb m1: It only has to handle the tasks pool and Intution pools. No problem for that.

Pepo stefanb: though it is sometimes possible to add new features for old apps, e.g. customizable gadtools look etc...

----- VOTE #1: Compatible OS, Existing API completely implemented ->

Binary compatibility for old hardware. New features are only available through NEW API calls or new Tags (if possible).

Pepo again that should mean the api should be _expanded_ not replaced.

----- RESULTS: 14 for, 4 against

griesbrei what's that thing with `old hardware` (sorry)

stefanb griesbrei: Amiga (TM) hardware.

m1elstv griesbrei, new CPUs cannot be binary compatible..

griesbrei I think we should`nt change _anything_ in the old OS

griesbrei yes, but everybody keeps talking on changing the tags in the old OS

Tau gries, not change, ADD

Murty griesbrei: I agree with you as far as that we first have to make something with the current functionality, and only add new features later (NEW tags, etc.)

griesbrei m1el: no, add new libraries or add new funtions to existing

stuff?

stefanb gries: Add new calls (and libraries if needed)

mlelstv griesbrei, compatibility is maintained by maintaining the old libraries.. simple and easy.

desrat pepo...the os as its stands is better than what they have

griesbrei some want to add new functions to the old libs, that`s not a good idea i think:

BM Hopefully, expanding them, wb.library would be fave.

Tau okay, in the first vote the decision was to retain and extend

the existing API by adding new calls and tags to the old

libraries whenever possible, instead of replacing whole libraries

stefanb BM: Kill WB :-) This would be a complete replacement, because it has NO features :-)

mlelstv griesbrei, depends.. a library should cover a specific topic. if

you say want to add new gadgets to gadtools or new features to

gadtools gadgets.. why not calling it gadtools.library ? why a

new one ?

Pepo Workbench needs a MAJOR overhawl, but this should be clear to anyone...

mlelstv pepo, workbench is one of the trivial projects!

mlelstv pepo, it`s really just an application

griesbrei mlelstv: again the problem with a new C=-made AmigaOS

Pepo mlelstv: still the nost important app :)

griesbrei mlelstv: if we change the existing libs, we won`t be compatible with new C= libs

mlelstv griesbrei, there won`t be new C= libs for long and if this

project gets really something done there has to be some cooperation.

Murty Since we are going to make an OS with all the old features

built-in (along with some tags added for new functionality) and

use all the new calls for new apps only, I think it`s safer to

make new libs for them.

jcompton If you want to be picky, there won`t be any more C= libraries.

griesbrei mlel, pepo: ok C= might not come back, but what if it really comes back??

griesbrei That would be a real mess with our OS

jcompton gries: Commodore will be liquidated, hopefully within a month.

Murty jcompton: One of the reasons we started this project in the

first place was that there may be no C= to give us new stuff, indeed. But I don't think we should exclude the possibility.
lemming Why not just add something to the begining/end of the replacement libraries

stefanb gries: A chance for our project and the Amiga, becuae WE already developed something that THEY can use. Remember 18 months before a NEW HW can come out.

Pepo griesbrei: if we implement most things C= would have done for 4.0, then there would be no problem for "them" (whoever) to join the work with us.

Arcade Murty: I agree with you, leave the old ones as they are ... add new features in additional ones.

mlelstv murty, the decision wether old or new lib should have nothing to do with new or old calls. libraries should cover some topic.
i.e. you should not make a new OO system for intuition but adjust BOOPSI

Murty If we add calls at the end of a library and so does C=, that's a sure-fire way to break programs.

mlelstv griesbrei, they have to if they want to make new Amigas :)

stefanb gries: If developers use OUR OS, then they have no real choice :-)

Murty stefanb: Yes, but so far the most likely to succeed Amiga people (not Escom, but C= UK) have been 100% uncooperative :-(

Lynet Murty: Have you received a "NO" from C= UK, or haven't they answered your letter?

Tau future Amiga hardware producers have no reason whatsoever NOT to coopearte with us

PieMan we've been going for 1:20 mins now.. do you want to have the second vote??

stefanb Yes, let's go on to the next topic...

Tau what IS the next topic?

stefanb Tau: Resource tracking?

Pepo resource tracking can be done easily for _new_ apps. it is impossible for old apps.

mlelstv proposes a poor-man's resource tracking

Murty stefanb: I think it's more important to get some real stuff done, like a design layout, let alone some code.

Pepo there just needs to be an object pointer in the process structure that automatically gets disposed when RemTask :)

Pepo Kudo: because old apps don't know about resource tracking.

Kudo pepo: hmm.. true... isn't there some free space? don't remeber
mlelstv pepo, RemTask() is dangerous..

Arcade Yeah, but how to do that? Make a call something like TrackMe()
to specify the object to be tracked?

Pepo mlelstv: ok, the internal DeleteProcess() from dos

Murty Resource tracking can be implemented quite easily, I think,
albeit only for new applications. They should specify what
resources can be freed and which are shared.

mlelstv pepo, means that you cannot kill a process easily.

Pepo mlelstv: kill = send a SIGF_ABORT signal

Arcade We would also need to add features to support the killing of
tasks more easily.

mlelstv pepo, a program would have to wait for that signal.

griesbrei We should create an AddRTObject(object,type,hook) call

Pepo mlelstv: right. resource tracking only for new apps

Murty mlelstv: When the user would want to remove a program (that has
crashed, for example, or is no longer wanted) the RT system will
be able to determine what can and what cannot be freed/closed.

Arcade Why a hook?

Kudo pepo: why not just remove the task from the list, and free the
resources?

Pepo every object _has_ a hook

mlelstv murty, even removing a task is dangerous, removing resources is
even more dangerous.

griesbrei the type specifies how to remove the object if the system knows
how to

Pepo Kudo: old apps _assume_ the resources won't get released, and
usually leave them allocated _on purpose_

Arcade Unless you tell the OS it's safe to remove it if the tasks fails
to.

griesbrei if the system doesn't know about it, let the hook do the work

Murty mlelstv: Removing resources is not dangerous when the task
expects it (i.e. when it set the MEMF_TRACK flag).

Lynet mlelstv: If new programs follow new rules, killing new programs
and freeing the resources would work.

mlelstv murty, still dangerous when the new program calls old libs..

Pepo griesbrei: there should be only one "type" of object.

mlelstv lynet, agreed.. so someone has to propose the new rules.

Murty mlelstv: No, because the old libs will allocate resources with

old flags. These resources will not be freed.

Pepo griesbrei: in other words, all objects can be disposed with the same call.

Murty (freed automatically by the RT system, that is)

Lynet mlelstv: I guess that would be a job for the design group.

mlelstv murty, means that each and every allocation needs to look for such a flag... hmm.

griesbrei pepo: yes, let's call it RemRTObjects()

Pepo griesbrei: no, it's called DisposeObject()

BM If an application is trashed it may no longer be looking for signals.

griesbrei pepo: it just calls internal routines for known objects and the removal hook for unknown

Murty mlelstv: No, why would every application have to look for the flag ? That's all handled by Exec.

griesbrei pepo: again, i'm not talking of OO

desrat pepo: why do i always wind up with less mem when i remove an application..particularly graphics

mlelstv murty, Exec ? No.. lets say a task opens a window. it needs to tell intuition that the window should be closed when the task is removed from the system. intuition needs to free associated memory, and so on.

griesbrei pepo: just an internal list handled by the RT system

BM CreateNewProcess includes a field for a cleanup routine to be called.

Arcade But if the task crashed, it could mean the cleanup fails too ...

mlelstv bm, where is the advantage of such a routine called by the OS and such a routine called by the application ?

Murty mlelstv: Ah, I thought of that :) The window will be added to the resources list with the TRKTYPE_WIN flag, which is defined in <memory/track.h>. The freeing routine will call

CloseWindow() on it. That's how it is in my head, anyway.

Pepo griesbrei: a list requires a node.

mlelstv murty, but each library, including user libraries can define their own objects..

griesbrei pepo: yes, the node will be created by the RT system

Murty mlelstv: Actually, that TRKTYPE flag is not hard-coded, it's allocated by Intuition. Oh, you just said that :)

stefanb Murty: Hmmm We shouldn't use static types. Every Object should

be registered by a special resource.library.

Murty AllocTrackType() in exec.library...

Arcade How about the new program calling the

DisposeOfThisObjectWhenImGone(void *window, TYPE_WINDOW); ?

Tau m1el, rather, I think, for programs with the TRACK_MEM hunk

flag, windows and other resources should be freed by default. If

something wants to leave a resource behind, it has to declare it

stefanb That's clearly OO :-)

PieMan with RT won't that in some way affect the compatibility?

Murty Tau: Makes sense, BUT then old programs would also have their
resources tracked (all of them). And that will cause trouble,

won't it ?

Arcade RT won't affect old (existing) programs if properly implemented

m1elstv tau, what about a simple callback function

(dos.library/AtExit()) ?

desrat stepahnb:sorry..temporarily lost my resource allocation...)

Pepo actually I'm not that convinced that RT is that necessary.

Apps need to free there resources anyway, when they exit cleanly.

Murty Arcade: Due to the sloppy use of the MEMF_PUBLIC flag, it WILL
cause problems. Believe me.

Tau murty, see the bit about the TRACK_MEM hunk flag, which would
only be set for new programs

Arcade pepo: But a task could always fail. If not by itself, then by
another corrupt one.

Arcade Murty: We'll leave the MEMF_ flags alone.

Murty Tau: I'd rather have it as a call, say when you open a certain
library than in a hunk.

Murty Arcade: Okay, and make a new AllocMem() for new programs. Fine
with me.

stefanb Pepo: Resource tracking would ease some parts of programs,
especially error processing...

Tau murty, there are piles of bits left for new AllocMem flags

Arcade Murty: Why the new allocmem? Just a call that says what type of
object it is, how to free it (if necessary) and the OS will do
the rest.

Pepo stefanb: _rely_ on the OS to do all cleanup?

Pepo stefanb: RT should not be (ab)used by apps.

Arcade Certainly not ...

griesbrei pepo: why not, at least you don't have to

Murty Arcade: Ah, you're going OO here :-) That's more or less what I proposed with the TrackTypes back there.

stefanb Pepo: Then RT is not useful at all...

Pepo a program which opens a file, and doesn't bother to close it is BROKEN and belongs in the trashcan

Murty Tau: There may be piles of bits left, but we don't want to get in conflict with any possible new versions of the OS, from C= or whoever.

Arcade People ready to vote for resource tracking now? (only usable for new programs.)

mlelstv stefanb, still, any kind of resource tracking would help debugging (you know what belongs to what process). And you can make structured exceptions..

Kudo arcade: why only new programs!?!

Tau murty, we're writing the next OS.

Arcade Kudo: cause it would break older ones ...

stefanb ml: Yeah, but without C++ they are most of the time a pain in the a**

Murty Tau: Yes, but just in case. It'd be silly to have programs break over things like that.

Tau murty, C= engineers are not stupid. If we do something that gains users, they will support it. Especially if it's something they'd done anyway

Arcade I've just yesterday posted a nice example to the mailinglist.

griesbrei tau: C= engineers not, but the managers..

BM C= for lack of a better reference. :)

stefanb Tau: I would propose this: RegisterObject(), AllocObject(), DisposeObject(). When you allocate an object using this functions, you'll get RT for free.

Tau gries, managers don't decide what bits will be used for AllocMem flags

Murty Tau: But why would we be a pain in the backside to the C= engineers, who'd have to work around our system to make both systems compatible, when we can easily avoid it ?

Arcade stefanb: something along that way, indeed.

griesbrei tau: but they decide where to go

Tau stef, it's fine except it requires an extra call for each object

Lynet stefanb: Then you'd have to register *all* resources to get proper RT.

Tau gries, they don't decide details like this

stefanb Tau: After AllocObject() you can do TrackObject(), and it will be added to the "automatic" free list of your task...

stefanb Lynet: That's the price...

Pepo when we fear that there might be a new (and different) OS from C= (or whoever), we could as well stop this project. Let's just ignore the possibility and continue (start?) working.

BM Ugh, open file, register file, close file.. Ick. :)

Arcade Okay, let's get on with the votes if that's okay with everybody ...

Murty Lynet: All resources would have to be registered, yes, but since most resources are allocated by library calls (CreateMsgPort(), Open#?()) they can do the tracking stuff

Tau pepo, yeah.. if there'll be a new OS from C=, why are we doing this in the first place?

mlelstv bm, not really. the dos.library would register the file for the process

Lynet stefanb: Ugh.. I wonder how many programmers are going to support this kind of cooperative RT.

stefanb BM: You allocate a FILE object, then you OPERATE on it, and after that you DISPOSE it.

stefanb BM: You can reuse an object...

stefanb Lynet: That's the price of a non-VM-system..

Murty Tau: There may never be. But we mustn't assume there WILL never be. It's short-sightedness like this that caused a certain business executive to say '640 kB ought to be enough for everybody'.

Lynet Murty: You're right. Make OpenLibrary() do RegisterObject(), and so on.

Arcade The problem with old programs is that a resource might be passed over to another program, so the resource tracking would free the object and leave the other program in the cold with unallocated memory. That's why resource tracking is impossible for old programs.

Tau murty, if we're next, their will support our version

Pepo Lynet: there would only need to be a single bit in a hunk that identifies a "new" program

griesbrei just some words

Tau I think we should find a solution where the OS automatically enables RT for applications marked as stuff that want it

stefanb VOTE #2: Resource tracking should be implemented for new programs.

stefanb - Proposed API:

Pepo stefanb: also, only high level resources should be "trackable".

It doesn't make sense to track a viewport, a colormap etc, when intuition tracks a screen itself...

stefanb - RegisterObject(): register a new object type for resource tracking

stefanb - AllocObject(): allocate an object

griesbrei leave that AllocObject away

stefanb - TrackObject(): Add an object to the automatic de-allocation list of the task

stefanb - DisposeObject()

Pepo stefanb: NewObject()

stefanb Pepo: There is already a NewObject()

stefanb Pepo: If you open a screen, Intuition handles the RT for you....

griesbrei RegisterObject and TrackObject are the same, aren't they=

stefanb Pepo: The proposed API is for the OS and for user-defined objects.

Murty Who needs NewObject() if you have AllocObject() ? You need a MakeObjClass() though.

stefanb gries: Nope, RegisterObject tells the OS, there is a new Object and how to allocate/free it.

Pepo Murty: ask the other way around: who needs allocobject() when we have newobject() (and we do have NewObject() since v36)

griesbrei Can be done in one call, I think

mlelstv stefanb, NewObject() has nothing to do with Intuition. It is just in intuition.library

Pepo strfanb: it would be a MakeClass() call actually

griesbrei TrackObject(objectHing,type,hook)

stefanb Pepo & ml: RT has to be on the lowest level. not in Intuition.

Arcade stefanb: we also need a call then to tell the OS not to track.

stefanb gries: NO.....

mlelstv stefanb, BOOPSI is on the lowest level. It is just misplaced in intuition.library

stefanb Arcade: Don't call TrackObject()!

Pepo Murty: you can call it how you want, but BOOPSI object system is completely generic.

stefanb ml: So we move the calls to Exec and the new Intuition uses the new calls!

Arcade stefanb: for programs that need to donate some resource to

another program and has that HUNK bit set.

Murty stefanb: What if the library call calls TrackObject() ? You need a tag that says TAG_DONOTTRACK, for a shared resource for instance.

mlelstv stefanb, I suggest a new library that covers the objects and support calls from amiga.lib

Tau well, that's a minor detail. Shall we vote?

stefanb Arcade: If you don't call TrackObject() it's NOT tracked.....

stefanb ml: Exec seems fine to me...

Pepo ATOM - Amiga's Tagged Object Model, an enhancement of BOOPSI :)

Tau stef, but the hunk bit makes everything tracked by default, thus a DontTrack()

stefanb Tau: No, you have to state when you want to have a Object tracked. The flag only states that you want your resource automatically FREED!

Tau stef, good point

Pepo I don't see the difference (read: advantage) over MakeClass(), NewObject(), and DisposeObject() as we have them now.

----- VOTE #2: Resource tracking should be implemented for new programs. Proposed API:

stefanb - RegisterObject(): register a new object type for resource tracking, tell the OS how to allocate and free it.

stefanb - AllocObject(): allocate an object

stefanb - TrackObject(): Add an object to the automatic de-allocation list of the task. Automatic de-allocation will only be used by programs which have a special HUNK bit set.

stefanb - DisposeObject(): Free an object.

stefanb - If a new program has a special HUNK bit set, it will use automatic de-allocation.

----- RESULTS: 19 for, with 5 against the proposed API

Tau API details will have to be worked out at implementation point in a smaller group

Murty Tau: ...called the design team :-)

Lynet Whow. We agree on *one* thing. Perhaps there's hope for this project anyway. :-D

Arcade topic: multiuser security

Tau arcade, filesystem level or memory? we already have the first, the second is not a personal computer OS issue really

mlelstv thinks the proposed API is a misnomer.. there are no objects involved.

Arcade Mostly at the filesystem level. If we could add some more, that would be a bonus.

Pepo security? that sucks, slows things down, and isn't used by

anyone.

Murty Tau: Filelevel system, please ! Memory level is one hell of a job and it requires a drastically different API from the current one.

Tau arcade, we HAVE filesystem level multiuser, so that's a non-issue

Tau murty, right, for one, it needs 100% MP

Murty As for MP, I didn't think the pools idea was bad. Again, new programs could automatically use protected memory and specify when they wish something to be shared.

Pepo Open() would just fail like with read protected files.

Arcade Is there? It needs to be supported within the OS, not by some external util.

Murty Like MultiUser is.

Pepo Arcade: the point is: there needs to be no redesign to support it. mlelstv murty, the OS interface is problematic.. lots of unprotected stuff. But, in any case, the probability for trashing memory is reduced a lot. And that's why even partial MP is good.

Tau pepo, as it does already, if the filesystem looks at the protection bits

Arcade But the system does require additions ...

mlelstv tau, that's a filesystem issue. of course one should support ownership in the filesystem.

Pepo things that don't need a new API should of course go in, if they make sense

BM Is there anything that could be done to make multi user file system MUFS less of a hack and more secure?

Murty mlelstv: Yes, many things just can't be protected in the current system, like library bases etc.

Tau mlel, that's what I said ;)

mlelstv murty, library bases can be write-protected most of the time

Arcade Yes, there are certain things that could make it more secure.

Pepo like e.g. a Network filesystem (envoy)

mlelstv tau, just wanted to support you :)

mlelstv murty, to get reasonable speed one would need 'aliases'. You do not want to change the MMU all the time.

Murty mlelstv: No, the changing of MMU pages was one of the things that concerned me in MP.

Arcade So, do we include multiuser security then?

Tau I think we should have a vote about the protected memory pools.
it has been discussed at some length already, so this should be quick

Murty mlelstv: It's too slow if you have to change them for every task, so the tasks would even have to share the same protected pool (?)

Arcade Let's have a proper vote anyway about a muFS-alike system.

Tau arcade, of course. Not including it would be dropping features, which I thought we decided not to do with the vote #1 ;)

mlelstv arcade, we cannot include security for local processes. the machine has to be a single-user machine. but it can provide multiuser services over a network.

mlelstv murty, no. each task has its own MMU table.

Tau everyone is for a filesystem with multiuser support, so there's no need for a vote

mlelstv murty, overhead for that is not that high..

mlelstv murty, basically you have to reload the ATCs at every task switch.

Arcade okay, voted unanimous then for multiuser-filesystem.

mlelstv murty, MP pools could even work for many old programs !

Murty mlelstv: How intensive is that ? How much relative time does it take, compared to just a task switch ?

Tau mlel, which is a nice extra ;)

mlelstv murty, you lose a few microseconds per taskswitch..

Pepo an MMU table switch is a write to a processor register.

Arcade Can someone in short enlighten me on this subject? (MP using pools)

mlelstv murty, the scheduler just sets the root pointer. the MMU will then have cache misses.

stefanb Vote #3: Add a memory protection flag for Pools (which will be ignored on machines without MP).

Murty Pepo: what about continually updating and changing the MMU tables ?

Kudo knows nothing about MMU's and MP pools... =(

Pepo the overhead comes with maintaining shared trees when there would be virtual address spaces.

Pieman not many people have mmu's anyway

griesbrei stefan: add a PUBLIC_READ and a PUBLIC_WRITE flags

Tau murty, it will add some overhead to AllocMem/FreeMem, but since it's using pools, they are not called that often

stefanb Piemann: They will have :-)

mlelstv arcade, you do not want to protect individual allocations (overhead). so you protect pools. pools (one or more) can live in a memory region made of MMU pages. Each set of associate pools has specific protections.

Kudo pieman: not many ppl have more than 640k...

stefanb Maybe each new program should get a MP pool automatically?

Lynet On the topic of MP - is it possible to protect the OS, or parts of the OS?

Arcade And what exactly is a pool? Is it private to a program, or can any process allocate from it?

mlelstv stefanb, that was my proposal.. each process has its own private pool and AllocMem/AllocVec automatically use that pool!

griesbrei a pool is private to the process

Tau stefanb, yes. Every program should get a private pool on NewTask(), whether they use it or not is their own problem

Pepo stefanb: for new apps non-public AllocMems() could use a system-provided private pool.

mlelstv griesbrei, not necessarily.. you can allocate pools to be shared.

Murty mlelstv: Doesn't the pool idea imply that there will always be a part of the memory that is protected and a part that is not ?

You'd add another CHIP/FAST division. What if unprotected memory runs out while there is still some protected left ?

Tau an old program would not allocate any memory from the protected pool, but instead use the shared pool for everything

griesbrei Tau: ..and add something like `This one needs 1Megs of RAM` like in Windoze??

mlelstv murty, it's not a fixed division, it is just an attribute reflected by the MMU settings

Pepo Gee, My Amiga ran out of protected memory!!

stefanb Murty: No, the memory in one pool is always of the same type.

Tau gries, no. Pools get new memory allocated to them dynamically when they run out of free space. You just allocate memory in larger blocks

mlelstv murty, when a pool needs another page it simply gets the same attributes

Pepo protection is not a "type". Any (non-chip) memory can be protected.

Pieman yes but it is possible that a program will need an increasing amount of memory, ie in mosaic storing the pictures

stefanb Piemann: But the will automatically shrink too...

Tau pepo, you'd run out of shared memory first. Every task would have its own private, protected address space

Murty mlelstv: I understand that. But if you have a large pool of memory (say 1 MB) that is protected and another one (4 MB) that is unprotected. An old program will use shared memory, so fill up the 4 MB. What if they run out while there's still 1 MB not being used ?

Tau gries, when it does the first AllocPool(), the pool is grown to, say 64k. When that runs out, another 64k is allocated.

mlelstv murty, the pools are allocated by the programs.. there won't be (much) free space in any pool.

BM With memory protection, will it be possible to keep the API?

Will addresses still be returned in messages to determine what gadget was clicked on for example?

Arcade Let's say the pool is at the end of allocated memory, how would it be able to expand?

Pepo Arcade: the MMU can remap

mlelstv arcade, just it expands now.

mlelstv arcade, a pool is made of puddles

Tau I posted a long article about this to the mailing list some time ago. Mlelstv commented on it, but I don't know how many of you saw it. I can send that if you want to see it

Pieman i guess it expands and shuffles the other pools along

Murty But if a pool grows dynamically, there will still be a lot of overhead when allocating, wouldn't there ? Wasn't that what we tried to avoid ?

mlelstv BM. that's the reason why MP cannot be 100%.

Tau stefanb, but the protected pool can be in private address space, and thus put in virtual memory

Pepo Pieman: no, that's what the MMU does. it translates addresses without copying the actual data.

mlelstv pieman, pools still are subject to fragmentation (within the pool). But most pools are per-process, when the process dies the fragmentation is gone.

Pieman pepo: so it's a bit like a fragmented harddisk..

mlelstv tau, any pool can be in virtual memory.

Pepo Pieman: no, seeking in memory doesn't take time :)

Tau murty, every time the pool is grown, the MMU table has to be

adjusted. Which is why it is a pool, instead of forcing an adjustment on every memory allocation

Pepo rozga: VM already works, doesn't it?

Pieman where is this mmu table stored? does the mmu unit have it's own memory or do we have to allocate it a pool of it's own?

Murty rozga: Not yet, but it will probably also be like the RT solution, only for new programs that is. I emailed the author of VMM and he said VM with the current OS was a bit of a hack :-(
Tau this pool stuff would make it HEAPS easier for VMM to operate
stefanb Pieman: That's the task of the OS...

Tau murty, because it has to hack in functions that are not designed for it.

mlelstv pieman, it's probably not a pool as it has to reside in its own pages.

stefanb Piemann: But probably write protected by the OS...

Murty Tau: So, the pool will grow a lot. If you make a puddle of, say, 64 K, then for every 64 K of memory allocated, a MMU table would have to be rebuilt. Sounds like a lot of overhead to me !

Lynet Tau: With pools, VM would be implemented by swapping out/in an entire pool?

griesbrei lynet: the puddles of the pool

Pepo Lynet: no, a page.

stefanb Murty: Less overhead than with MMU adjustment every AllocMem()

mlelstv pieman, there is a lower level that handles regions of memory with certain attributes. The MMU table will use its own region
BM I think all these shareware authors should be contacted for their opinions on what hacks they had to perform to get their code to work.

Pepo Lynet: a page of a pool. every page could hold many puddles (allocations) though

Murty Tau: No, mainly because the MEMF_#? flags are abused by many applications and it's very hard for VMM to determine what it can cache and what it can't. System structures, for example, cannot be cached, so VMM looks if the allocation is smaller than, say, 500 bytes before writing disk

Tau lynet, no, a page at a time. but the VM table need to be updated only when a pool grows/shrinks

mlelstv pepo, means that you want multiple pools per page.

Tau murty, that and lack of pools

stefanb Murty: MP is only for new programs. VMem too...

mlelstv tau, not even then.. just when the pool needs more/less pages

Murty stefanb: I know that. I was just explaining that :)

Tau murty, protected pools would make VMM magnitudes simpler

Pepo mlelstv: if they have the same attributes, who not?

mlelstv stefanb, MP+VMEM can work, often, for old programs too

Tau mlel, I'm assuming puddle size > page size

mlelstv pepo, I did propose that :)

Murty Here's my suggestion : use the pools for VM, but let's forget about MP. It's probably going to be kludgy anyway, with so much stuff unprotected.

stefanb OK, what should we vote on? Just "We want to implement MP (and VMem) for memory pools"?

mlelstv tau, well, yes.. but with 8K pages this might not be the case :)

Arcade tau: better puddle size = $X * \text{MMU page size}$

stefanb BM: Machine dependend. The puddle size will be adjusted by the OS.

mlelstv arcade, or $1/X * \text{MMU page size}$

mlelstv stefanb, the puddle size is suggested by the application. it might be adjusted though.

Tau mlel, no point making it smaller than MMU page.. you reserve the whole page anyway

Pepo mlelstv: a puddle is what you get returned from AllocPooled()

stefanb ml: That's what I meant... The same is already true for AllocMem().

Murty Here's my suggestion : use the pools for VM, but let's forget about MP. It's probably going to be kludgy anyway, with so much stuff left unprotected.

mlelstv tau, a program should choose puddle sizes for specific objects (to minimize fragmentation). a task may have many pools and not all of them will require their own page

Tau murty, MP is a VERY VERY nice thing to have. and a VM API is so close to MP API there's no point in not having both

Tau mlel, I'm talking of the task global MP pool, tho

caldi Murty: Perhaps someone should contact Mike Sinz, according to the devcon notes, C's VM plans were pretty far along. They too planned VM in pools with the PAGED attribute.

mlelstv tau, well.. that's not necessarily what we called a 'pool' :)

Murty stefanb: Well, kludge is not the right word. The Dutch word I'm looking for is 'halfbakken', which is something like

half-decent. You can only protect half your resources, but if the unprotected half is corrupted, chances are that the rest will go down with it.

Tau murty, half-baked is an English phrase, too ;)

Arcade murty: exactly. MP can never be safe

stefanb Murty: But better this than nothing...

Tau arcade, MP is inherently safer then no MP

mlelstv tau,arcade: safe but not secure

Murty Tau: But is half-baked :-) protection worth the effort ?

stefanb Arcade: The program will still work though, because the OS ignores the MP flag on non-MP hardware.

Tau murty better than none at all

stefanb Murty: You aren't a programmer, aren't you? :-)

Tau stefan, we should also make it optionally ignore it on user request (an old program kludge)

Murty stefanb: I am, actually. Why ?

----- Vote #3: Add memory protection to memory Pools.

stefanb - MP will be ignored on non-MP hardware

stefanb - MP can be switched off if the user wants.

stefanb - New programs automatically get an MP memory pool for their purposes.

----- RESULTS 15 for, 1 against

----- Vote #4: Same points for Virtual Memory.

----- RESULTS 15 for

mlelstv griesbrei, the most probable thing is that while parts are rewritten they get new features on the way.

Lynet Pepo: Which new features has been added to intuition-clone?

Murty One topic I'd just like to mention is the speed at which is project is, or rather isn't, moving. Does anyone have a brilliant idea to speed things up on the designers list ?

caldi Murty: Is there a designers list one can read over, perhaps commit to some part of the project?

Pepo Lynet: not much yet: window hiding, realtime resize and drag, several bugfixes, a new gadget method, new multiple pen array support for borders and some other small things (mouse acc etc)

Lynet Pepo: Sounds nice. Is it soon ready to distribute to betatesters?

Pepo Lynet: I expect to have a working _alpha_ version in 2 months. several old (v40) things are not done yet (e.g. no sysreqs, no custom pointers, no sysiclass, ...)

Lynet Pepo: Ok. Can't wait to start betatesting. :)

Pepo lynet: alpha, not beta. beta versions to be released in perhaps

8 months

Pepo Tau: alpha testers will be programmers only, not users.

Tau pepo, programmers of intuition?

Pepo Tau: yes.

1.10 The Heart of Installer

Installer Basics - Intro & Tool Types

By: Robert Reiswig - rcr@netcom.com

Installer Article?

When I was first asked to do an article on Commodore's Installer I was not sure if I wanted to do it. Everyone thinks that the installer is some small little scripting language that anyone could whip up in a few minutes. Well if you have looked at an installer "script", you will see that it has its own language and takes more than a few minutes to come up with a nice installer for a program. After a bit of nugging by "Wubba", (hummm, wonder what his real name is? Or is he just a bot :) I was convinced to put it together.

Installer Basics.

I'm not sure on what I will cover, but my idea is to give you the basics to writing your own install for your program. Things such as asking questions, if then's, copying files, the basics.... Maybe I'll pick a simple program off Aminet that needs and install, and give the steps needed to complete one.

Installer Intro

As more and bigger programs come out for the Amiga, the number of different ways to get these programs off of the floppies and on to the Hard Drive kept growing and growing. Some companies had you drag icons around while others had you run their installer that never quite worked right. This did not go over well with most end-users. Things such as referring to the Hard Drive as "DH0:" or requiring 1.3 of the Operating System to use the installer.

Around the beginning of 1992 Commodore started to look into this problem. They ended up with a programming/scripting language that was based on LISP. The language is not that hard to use, you just have to get use to the large about of "(" and ")" that you must use. I'm not sure on what percentage Commodore actually programmed, but if you do an "About..." when you first run an install script you get information on who and what the install is for and about.

This allowed programmers , and their companies, to give the end-users an install that would look and feel the same from one application to the next. In return giving the program a more overall "professional" presentation along with the end-user not having to relearn a new Installer.

NOTE: If you do not already have the "Installer" program in your "c:" directory, it would be a good idea to get your Install Disk (the one that comes with the Operating System Disks), and copy the Installer from there to "c:!" At least place it somewhere in your path.

Tool Types

The following is a list of Tool Types that you can use for the Installer with example uses.

Most of the installers I do are for AmiTCP/AS225/TIA/Mlink programs. Therefore the NOVICE & AVERAGE users don't really apply. I try to write the install to be as straight forward as possible, to avoid having 3 levels of users. I normally use only the APPNAME, MINUSER and DEFUSER. I let the end-user set the rest (within the installer) when they run the installer script.

APPNAME [APPNAME=Installer Basics]

This is used too set the Program Name that you are installing. You must set this! It will show up in the opening window when the Installer is run. "Welcome to the Installer Basics installation utility."

MINUSER [MINUSER=EXPERT]

There are 3 options you can set MINUSER to: NOVICE, AVERAGE, EXPERT. I don't use these to much but you can set MINUSER to a setting and then within the installer script check these and act on them.

DEFUSER [DEFUSER=EXPERT]

If you set MINUSER to NOVICE or AVERAGE then you can set DEFUSER (Default user) to what you think would be the most common lever of user for your program.

NOPRINT [NOPRINT=FALSE]

If NOPRINT is set to FALSE, then the "Printer" option will be marked out. (This does not work on some versions of the installer.)

PRETEND [PRETEND=FALSE]

If PRETEND is set to FALSE, then the "PRETEND" option will be marked out.

LANGUAGE [LANGUAGE=german]

This is very useful when you are doing an installer for more then one language. You can have many different icons (one for English, German, ect..) Then within the installer script you can check the variable "@language" and then use the correct language in the installer. "english" is the default.

LOGFILE [LOGFILE=RAM:LOG.txt]
This is where you can tell the Installer the name and where to put the log file if the end-user wants one created. "install_log_file" is the default.

LOG [LOG=FALSE]
If the user picks the NOVICE mode (and you let them pick it) the default is to create a log. If you have LOG set to FALSE then it will not create the log in NOVICE mode.

SCRIPT [SCRIPT=scripts/english]
If the installer script is located in a different directory or is named different than the ".info" file, you can use the SCRIPT var to tell the installer what script to use. This is nice if you have many scripts and you don't store them in the root dir.

As with all ToolType encasing them in "()" will turn them off, this is useful for testing. (ie "(LOGFILE=RAM:LOG.txt)")

Well that is about it for now ... hoped you learned something!

If you wish to get more detailed information there is an Installer archive in/on the Fred Fish Collection.

This article is ©1995 by Robert Reiswig
If you wish to reprint this, all you have to do is ask. Also if you have any questions, please let me know! I can be reached at rcr@netcom.com

1.11 Methods to the Madness

BOOPSI OVERVIEW

~~~~~

This article is designed to provide the novice BOOPSI programmer information to aid in writing a custom gadget class for an application. The reader is assumed to be familiar with some basic BOOPSI and/or OOP concepts. I recommend reading the BOOPSI chapter of the RKM:Libraries Manual, 3rd Edition, then perhaps come back and read this overview again and it should start to make more sense.

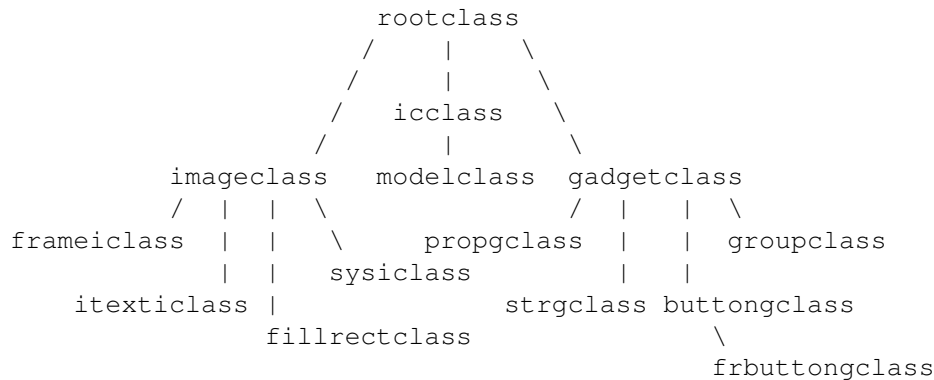
In BOOPSI, everything is an "Object". Each object is a distinct entity. Certain objects have common characteristics and can be classified in to different groups called classes. As objects, "Mehdi Ali" and "Irving Gould" are all distinct objects, but they all have something in common; a year ago they could all have been classified as "Over Payed", today, I guess you would classify them as "Unemployed". A specific object is an "instance" of a particular class, for example, "Dork" is an instance of "Mehdi Ali".

Classes may be divided in to subclasses. A fruit object can have several subclasses such as apple, orange, grapefruit, etc. The inverse relation is also true, fruit is a superclass of apple. Additionally, "Over Payed" and "fruit" are in fact subclasses themselves of a universal category typically called root.



## BOOPSI CLASS HIERARCHY

~~~~~



In BOOPSI the root category is called "rootclass". Intuition has three subclasses of the rootclass: gadgetclass, imageclass, and icclass. All boopsi gadgets in the system are an instance of gadgetclass, and all boopsi images are an instance of imageclass. A new concept to Intuition is that of interconnection, or the icclass. Interconnection allows a group of gadgets to "talk" to one another, to notify another object when some specific event occurs. For example, a scroller gadget might be interconnected to a listview gadget such that the scroller gadget controls the position of the listview.

Under imageclass, we have "frameiclass" which does your standard raised or recessed bevel box. Under V39 or greater there are a number of bevel styles available. The "sysiclass" contains a number of standard images mostly used by Intuition, specifically, the front/back gadget, close gadget, arrows, and checkmark imagery to name a few. The fillrectclass is often underused, it does what its name implies, it renders fill rectangles. I suspect a direct call to graphics.library's RectFill() or EraseRect() would be significantly faster.

Under gadgetclass, propgclass creates proportional scrollbars, strgclass creates a string gadget and buttonclass creates a button gadget as you might have already guessed. At this point, these classes are not important to us and will not be discussed.

DESIGN DECISIONS

~~~~~

When designing class, you must first decide which "superclass" is suitable for your needs. When creating an image class, the "superclass" will be either "imageclass" or some subclass of "imageclass". When creating a gadget class, the superclass will typically be "gadgetclass", or one of its subclasses. The example provided will concentrate on creating simple subclass of "gadgetclass".

Your custom classes may be public or private. Any application can access a public class, and typically a public class will take the form of image or gadget shared library. A public class has a name associated with it, such as "listview.gadget". Private classes are typically compiled and linked with an application. A private class has no name associated with it, so without a pointer to the class, it can not be used. As you read on, you find it is possible to create private classes that are publically accessible shared libraries by using a simple function you

can supply in your class library. The advantage the later approach is that you do not have to worry about finding a unused name for the public class. Currently, there isn't any means by which you can register a public class name, and it wouldn't be good if two incompatible classes used the same public class name.

#### OBJECT METHODS

~~~~~

In an OOP system, an application requests an object to perform some method (read function/action) by sending the object a message. BOOPSI also passes methods via a message passing system. This is not the same as the Exec message passing system, so do not get them confused.

The follow methods are common for all classes of objects;

OM_NEW

Simular to calling `CreateGadget()` for a GadTools gadget, in BOOPSI you call the `NewObject()` function to create an instance of a pariticular gadget class. At this point the `OM_NEW` object method will be invoked on that class. In turn the class will pass this method to its supermethod (via `DoSupperMethod()`) which eventually will works its way to rootclass which will allocate memory for an instance of the object. The instance data is usually some structure whose fields hold per instance data for a given class. After calling the supermethod to allocate an instance of your class, the class can set defaults, parse any special creation tags and prepare the object to be used.

From within `OM_NEW`, your class might also `NewObject()` other objects to make some type of composite gadget, or perhaps just use an image class to render various bevel or group boxes. A listview is an example of a composite gadget. Its made up of the scroller area, a proportional scroller and two buttons with arrow images.

Should you fail to allocate the classes you need, you must make sure the `OM_NEW` method to your class returns `NULL`. But before doing so, you must use `CoerceMethod()` in a manner simular to calling `DoSuperMethod()` in order cause an `OM_DISPOSE` method to be invoked on yourself so the previously allocated instance data will be released.

OM_DISPOSE

In GadTools, you call `FreeGadgets` to free all gadgets created with `CreateGadget()`. In BOOPSI, you call `DisposeObject()` ON EACH INSTANCE of any classes that have successfully returned from `NewObject()`. This will invoke the `OM_DISPOSE` method on the class, for most class writers you can safly pass the `OM_DISPOSE` message to your super method directly. However, if you have any extra memory, or classes to close you must do proper clean up first, then pass the message on to `DoSuperMethod()` which will free your instance data allocated thru the rootclass.

OM_SET

Once a gadget is created via `OM_NEW`, you may wish to change certain certain aspects, text labels, pen selection, etc. This can be accomplished via `SetAttrs()` for images and `SetGadgetAttrs()` for gadgets.

These calls will invoke an OM_SET message on the specified object instance. The object should always pass this message to its super class first, then if ops_GInfo field in the (struct opSet *) set method message is non-null, you may check any special tags you allow to be altered after the gadget is created. If you require that your gadget be rendered again after OM_SET, you must return a 1, otherwise return 0. In turn this value is returned to the application which should then refresh the said gadget with one of the gadget refresh functions.

OM_UPDATE

Very much the same as OM_SET, in fact many classes reuse the same sections of code for OM_SET and OM_UPDATE. The main difference being that OM_UPDATE typically causes an object to update various values and in the case of a gadgetclass, you might cause the gadgets visual appearance to update to reflect the new settings. A class typically receives OM_UPDATE methods when another object is interconnected via notification, such as a scroller updating the position of a scrolling list.

OM_NOTIFY

To support interconnection, to need to send yourself an OM_NOTIFY message typically via DoMethod(). This method needs to set up a tag list consisting of the GA_ID of your gadget, and any of the tags/values you wish to notify the interconnected objects of. This tag list is then passed to your superclass via DoSuperMethod() thus causing an OM_UPDATE with the said tags to the interconnected objects almost similar to programmatically updating them with SetGadgetAttrs() or SetAttrs().

GADGET METHODS

~~~~~

The follow methods are common for all classes of gadget objects;

#### GM\_HITTEST

For rectangular box gadgets all you have to do is return the value GMR\_GADGETHIT as defined in <intuition/gadgetclass.h> which verifies the mouse click is indeed within your gadget's "hitbox" dimension. For a simple button, the hitbox is the clickable area which makes up the button including any outer bevel box. If your gadget is not a box, you will have to test the mouse X/Y position of the mouse click provided in the GM\_HITTEST message to determine if it is inside your clickable region of the hitbox. If it is, return GMR\_GADGETHIT, if not return a 0. Remember a hitbox is always rectangular areas, if you choose to make a round button the round area must be contained within the bounds of the hitbox rectangle.

#### GM\_GOACTIVE

Assuming you returned GMR\_GADGETHIT from GM\_HITTEST, this is the next method you should receive. This method allows the gadget to become the active gadget, ie, the gadget that will gain the input focus. This method should return GMR\_NOREUSE if it does not

want to become the active gadget, or GMR\_MEACTIVE if it does.

In the case of a listview, you might test the mouse position for a hit on a particular node in the list, and/or cause an update the visual state of the gadget in to a selected mode by sending yourself a GM\_RENDER method via the DoMethod() function.

#### GM\_HANDLEINPUT

Once you have become the active gadget, you will receive all mouse, keyboard and intuitick input. Based on the mouse location you can update your gadget settings and cause a visual render update to occur by sending yourself a GM\_RENDER method via the DoMethod() function. This method can be one of the mose simple, or one of the most complex depending on the gadget type. For most classes, the grunt of the work will be in the GM\_RENDER and GM\_HANDLEINPUT methods.

#### GM\_GOINACTIVE

This gives the gadget an oportunity to do any final calculation or rendering after the gadget has lost the input focus, such as rendering the gadget in the non-selected state via sending yourself a GM\_RENDER method via the DoMethod() function.

#### GM\_LAYOUT

This method was introduced in V39 (AmigaOS 3.x). It is invoked when a GREL gadget is first added to a window either with AddGLList() or gadgets present in a window at window open time via the WA\_Gadgets window attribute tag. It will also occur when window resize takes place. This allows the gadget to lay itself out based on the new window or requester size found in the GadgetInfo structure present in the layout method message. You must be careful not to do any rendering inside this method or a C= engineer will personally come to your house and beat you silly. Instead, Intuition will send a GM\_RENDER when it is time to (re)render the gadget. If you wish to remain V37/V38 compatible you should not rely on this method.

#### GM\_DOMAIN

A somewhat brain-dead means by which a gadget layout/group class can determin a gadget object's minimum, nominal, and maximum sizes. Typically, only minimum needs to be implemented for a good layout. Infact, to date, I'm not aware of many public classes that requiers it. This is a method that is new to V39 (AmigaOS 3.x) and higher. Intuition does not make use of it and the current AmigaOS 4.0 effort may not make use of it either, but instead provide an alternative method to layout font sensitive GUI.

#### GM\_RENDER

Here, you are to render the imagery for your gadget by using the various image subclasses of imageclass, and/or some of the simpler graphics rendering functions. Keep it simple here, your executing on Intuition's task. Try to limit your rendering to utilizing image classes via DrawImageState() or DrawImage(), or simple rendering routines

---

such as `Text()`, `TextLength()`, `TextFit()`, `Move()`, `Draw()`, `RectFill()`, `EraseRect()`, `BlitTemplate()`, etc.

Within the `GM_RENDER` message, the `gpr_Redraw` field (for definitions of the method message structures see the `<intuition/gadgetclass.h>` include) will be one of three values:

`GREDRAW_REDRAW` - (Re)draw entire gadget.  
`GREDRAW_UPDATE` - The user or application has manipulated the gadget in some way, only render the imagery needed to reflect that change. Example, moving the level handle if a slider gadget.  
`GREDRAW_TOGGLE` - If the gadget is a boolean toggle type, toggle it to or from the highlighted imagery.

You may call this method from another method, such as `GM_HANDLEINPUT`, to cause intermediate updates to the gadget imagery according to how the user moved the mouse or the keys that were typed in.

BEAM ME UP SCOTTY

~~~~~

By now either I totally confused you, or you have a basic feel for how the BOOPSI system works. Next month we will cover the `imageclass` specific methods as well as expand on the information provided here and create a working public class as well as a test program to demonstrate the function of the class.

Christopher Aldi can be reached at caldi@pcnet.com

1.12 Revision Control and SMAKE under SAS/C

USE OF MAKEFILE IN PROJECT MANAGEMENT

Preamble

A surprising number of programmers are not aware of the various tricks that can be used to simplify the compilation of their programs. A Makefile can play a big part in this process, but many times it is sadly neglected or sometimes not in fact used at all.

Even a larger number of programmers have their source files in varying degrees of chaos, with only random copies, if any, of the older versions for reference or backup.

Most programmers have at one point or another come across the situation where an archive of their source code would have saved a lot of trouble. You might have accidentally destroyed a source file or even the whole source tree of your latest program, or maybe introduced a bug that you just can not track down and fix, but which you know wasn't in the last week's version.

Revision control is the answer to these situations. In this article I will try to explain how to use both Makefiles and revision control systems to help development of your projects.

The environment

I developed the techniques described in this article over a couple of programming projects. The development platform used is an Amiga 3000, and the software consists of the SAS/C 6.51 compiler, Heinz Wrobel's port of the RCS software (HWGRCS) and Many features of the SMakefile used as the project base are unique to SAS/C's SMake, and it very probably won't work without some changes in other environments. The ideas, however, are global, and this article should be able to give you some hints even if you use another compiler.

Makefile

Make is a tool that operates on files using a set of rules as the guide to what to do to each of them. These rules are listed in a file called the Makefile. In essence, a Make rule consists of a target file, a list of the source files on which it depends, and the instructions on how to create the target file from the source files. A simple example might look like this:

```
foobar: foo.o bar.o
    slink foo.o bar.o to foobar
```

This is one Make rule. It says "to create 'foobar' from 'foo.o' and 'bar.o', run the command 'slink foo.o bar.o to foobar'". Make will then look for the source files and determine if 'foobar' needs to be updated or not. Make also has some built in intelligence. For example, it knows that if it does not find a file with the name ending in '.o', it should look for the corresponding file ending in '.c', and compile that file first. Thus, 'foobar' gets updated if either of the two source files 'foo.c' or 'bar.c' has been changed since the last update, and only the parts that need to be compiled will be.

RCS

RCS, short for Revision Control System, is a software package used to manage multiple versions (revisions) of files. It facilitates the storing, retrieval and merging of versions of the same file, letting the user log the changes. It is useful as the storage system of various types of files that are revised frequently. Program source code is only one, although an obvious example.

RCS can also be used in an environment where many people use the same source tree, letting only one person have a modifiable copy of a file, thus limiting the possibilities of two persons making changes at the same time. This might sound complicated, but in effect RCS can be very simple to use.

The basic use of RCS is based around two commands, "ci" (check in) and "co" (check out). These commands let you store a file to the RCS database, and get a copy of that file out for viewing or modifications. You could, for example use "ci -r2.13 niftyprog.c" to store the source of version 2.13 of your utility program. While RCS is usable manually, it does get

somewhat tiresome as the project size grows. There is also a problem with keeping RCS revision numbers and your own program version number agreeing, unless you have some automated help to handle that.

Here's where a good Makefile comes in handy. A Makefile can automatically increment the revision number after each recompile, and when you are happy with the program, you can archive the current revision with the correct number with a single command.

The SMakefile

There is no single solution to the problem of creating a Makefile that can handle not only the building of a project, but also the archival of it. I will list here a stripped down version of the template SMakefile I build my projects on. The whole version, complete with comments, is included as a separate file. In the following, the lines are numbered for reference later in the article.

```
1.  # SMakefile for SAS/C
2.
3.  # Project files
4.
5.  PROG=      xxx
6.  SRCS=      $(PROG).c
7.  OBJS=      $(PROG).o
8.  INCS=      $(PROG).h
9.  MISC=      SMakefile
10. LIBS=
11. DEBUGLIBS= LIB:debug.lib
12.
13. # Files to delete on "smake clean"
14.
15. JUNK=       \#?.(o|map|lnk|gst) SCOPTIONS
16.
17. # Build options
18.
19. CFLAGS=     DEBUG=LINE OPTIMIZE NOSTACKCHECK
20. DCFLAGS=    DEBUG=SYMBOLFLUSH NOOPTIMIZE DEF=DEBUG
21. LDFLAGS=    STRIPDEBUG
22. HOOKFLAGS=  NOSTACKCHECK NOSTACKEXTEND
23.
24. # Compiler defaults
25.
26. SCOPTIONS=
27.
28. # Programs used
29.
30. MAKE=       smake
31. VER=        ver
32. CP=         copy
33. CAT=        type
34. RM=         delete quiet
35.
36. VERSION=    '$(CAT) VERSION'
37.
38. # Standard targets
```

```
39.
40. test:
41.     $(MAKE) "CFLAGS=$(DCFLAGS)" "LIBS=$(DEBUGLIBS)" "LDFLAGS=" $(PROG)
42.
43. all:      clobber $(PROG)
44.
45. # Build rules
46.
47. $(PROG):  SCOPTIONS version.o $(OBJS)
48.     $(CC) version.o $(OBJS) LINK TO @$ BATCH $(LIBS) $(LDFLAGS) $(CFLAGS)
49.     $(VER) # bump revision for next link
50.
51. SCOPTIONS: SMakefile
52.     $(CP) TO @$ <FROM <
53. $(SCOPTIONS)
54. <
55.
56. version.c:
57.     $(CP) TO @$ <FROM <
58. const char VersionID[] = "$$VER: $(PROG) " VERSION " " __AMIGADATE__;
59. const char Version[] = VERSION;
60. const char CompileDate[] = __AMIGADATE__;
61. <
62.
63. version.o: version.c $(SRCS)
64.     $(VER) -n # make sure VERSION exists
65.     $(CC) NODEBUG DEF=VERSION="$(VERSION)" version.c
66.
67. rcs:
68.     ci -l$(VERSION) $(SRCS) $(INCS) $(MISC) $(DOCS)
69.
70. clean:
71.     $(RM) $(JUNK) $(PROG)_Cat.c >NIL:
72.
73. clobber:    clean
74.     $(RM) $(PROG) >NIL:
```

Description of the SMakefile

In the beginning of the SMakefile are listed the files making up the project. The name of the program is assigned to the PROG variable, and all the object, source, and headers files are listed in OBJS, SRCS and INCS, respectively. MISC is for the other files of the project that should be archived, including the SMakefile itself. Link libraries used will be listed in the LIBS variable.

The JUNK variable contains the pattern or list of the files that are created during the build process, and can be deleted to make room.

On lines 19-22 are listed the options used in the various build stages. CFLAGS will be used when compiling the "final" version, while DCFLAGS holds the options for a "debug" compile. LDFLAGS is the option variable for the final linking. HOOKFLAGS contains the options needed for callback hooks (you will not want stack checking on here, even if you'd like it elsewhere).

The options that will be set all the time, regardless of the build mode, can be saved in `SCOPTIONS`. The `SMakefile` will automatically create the `SCOPTIONS` file from this variable.

Lines 30-34 list the programs used in various stages of the compilation or project management. By assigning these into variables, there is no need to repeat the actual program name in the `SMakefile`, and programs are more easily changed.

The `VERSION` variable will be set to the current program version during compilation.

Next is the default rule, which simply calls `SMake` again to build the project using debug settings. This way you can simply run `"smake"` without any options to build a debug version of your binary. Line 42 has the `"all"` rule, which is what you would use to create the final binary.

The rest of the `SMakefile` lists the standard rules for the project. These rules should not change from project to project. The included, larger `SMakefile` also includes rules used to create localized source from the catalog description.

The first rule creates the actual binary target file. It will simply link all the object files together, and if that was successful, bump the revision number up one so that the next compilation will get a new version. This is done using a small special utility `"ver"`, source to which is included.

The `SCOPTIONS` rule recreates the `SCOPTIONS` file if you change `SMakefile`. To rebuild all source files using the new options, add `SCOPTIONS` to the dependency list for each source file. This dependency list (as created by `Mkmmk`, for example) can be appended to the end of the `SMakefile`, or inserted anywhere within.

The next three rules are for creating the `VERSION` file that holds the build number, the `version.c` source file for the version strings you could use elsewhere in your program (use the `Version[]` variable for version number instead of hardcoding it in other source files), and for compiling this file.

All that is left are the three special rules for archiving the current source versions and cleaning up the object files and other `"junk"` cluttering up the directory. The `rsc` rule, on line 67, will store all the source and related files using the build number stored in the `VERSION` file. Note that this is one revision higher than the binary created in the last compilation. This is intentional, since the `RCS` operation will bump the timestamps on all the files. You would normally test the program using the debug version compiled with `"smake test"` (or simply `"smake"`), and then when you are satisfied with the operation, update the documentation etc. and call `"smake rsc"`. This will store the files and ask you for the log information for all of them, leaving a modifiable copy of the file (updated with the new log info) in the directory. You can now call `"smake all"` to compile this version, and it will have the same version and revision as the one archived in the `RCS`.

This article is Copyright © 1995 Osma Ahvenlampi, <Osma.Ahvenlampi@hut.fi>. All rights reserved. No part of the article may be copied or distributed in electronic or printed publications without prior permission of the author. Permission hereby granted to the Amiga Report Tech Journal.

1.13 smakefile

You'll find that, along with this guide file, a small archive was unpacked; this archive is "smake.lha", the necessary stuff to use the information given in the article. Thanks again, Tau!

1.14 A

From: Kasper.B.Graversen@pl.f36.n237.z2.fidonet.org
Newsgroups: comp.sys.amiga.programmer
Subject: Fell In Love with GOLDED
Date: Tue, 04 Apr 1995 02:10:04
Organization: News Server at UNI-C, Danish Computing Centre for Research and Education. ←
Lines: 128
Message-ID: <2f80e2bc-nntpgate@nrg.dtu.dk>
NNTP-Posting-Host: www.nrg.dtu.dk

```
?MSGID: 2:237/36.1 2f80a064
Hi Matthias...
You wrote about <Fell In Love with GOLDED>
```

```
>> Emacs? Isn't that the sucks editor that came with Workbench?
MS> No, that's MicroEMACS. Smaller and faster, but less features.
```

ahh ok...

MS> BTW anybody got the source for "windows-nt.elc" already? ;-)

is this what you're missing?

Take a look at this. Straight from MicroSoft ;-)

```

/*
TOP SECRET Microsoft (c) Code
Project: Chicago(tm)

```

Projected release-date:

Summer 1994^H^H^H^H^H^H^H^H^HSpring 1995

 $\ast /$

```
#include <stdio.h>
#include <dos.h>
#include <conio.h>
```

```
#include "win31.h"
#include "evenmore.h"
#include "oldstuff.h"
#include "billrulz.h"

/*
Reference:
Internal memo #99281-95 from:
    William H. Gates III
    to:
    Executive managers Chicago(tm)-project

William H. Gates III wrote:
"I have serious doubts about the 'EASY' installation-definition.
It might prevent customers to think that they actually bought something
_good_. Therefore I want the installation-definition to be 'HARD'.
```

Carry on,
God^H^H^HBill

```
"
*/
#define INSTALL = HARD

void main()
{
    while(!CRASHED)
    {
        display_copyright_message();
        display_bill_rules_message();
        do_nothing_loop();
        if(first_time_installation)
        {
            make_50_megabyte_swapfile();
            do_nothing_loop();
            totally_screw_up_HPFS_file_system();
            search_and_destroy_the_rest_of_OS/2();
            hang_system();
        }
        write_something(anything);
        display_copyright_message();
        do_nothing_loop();
        do_some_stuff();
        if(still_not_crashed)
        {
            display_copyright_message();
            do_nothing_loop();
            basically_run_windows_3.1();
            do_nothing_loop();
            do_nothing_loop();
        }
    }
}

/*
Reference:
Internal memo #99683-95 from:
```

Executive managers Chicago(tm)-project
to:
William H. Gates III

Executive managers Chicago(tm)-project wrote:

"Dear Sir,
Since we have found that this last piece of code within the
'if'-statement
will never execute, we descided NOT to include it in the final code.
This way we will save atleast another 5 megabytes of consumer-diskspace!

Thank you for listening to us,
the executive managers of the

Chicago(tm)-project

```
"
*/
/*
    if(still_not_crashed)
    {
        write_cheer();
        finished();
    }
*/
    create_general_protection_fault();
}
```

=====

```

_____ . . .
(____ :|/ _____ \_
|    /  _/ | : /  _// _____ . .
|    \  ~\_| _\ .\ _ / _ : | _
| _ _ | \____/ | _____/
! || |=====
: || | -+ Kasper B. Graversen +-
. || :
```

... UserInfol.00:CGIAJGFIBBIDFEECAFEHILDBHGDDBLCGJ01

--- Spot 1.3a #604

* Origin: Even Russians use UserInfo - why don't YOU? (2:237/36.1)