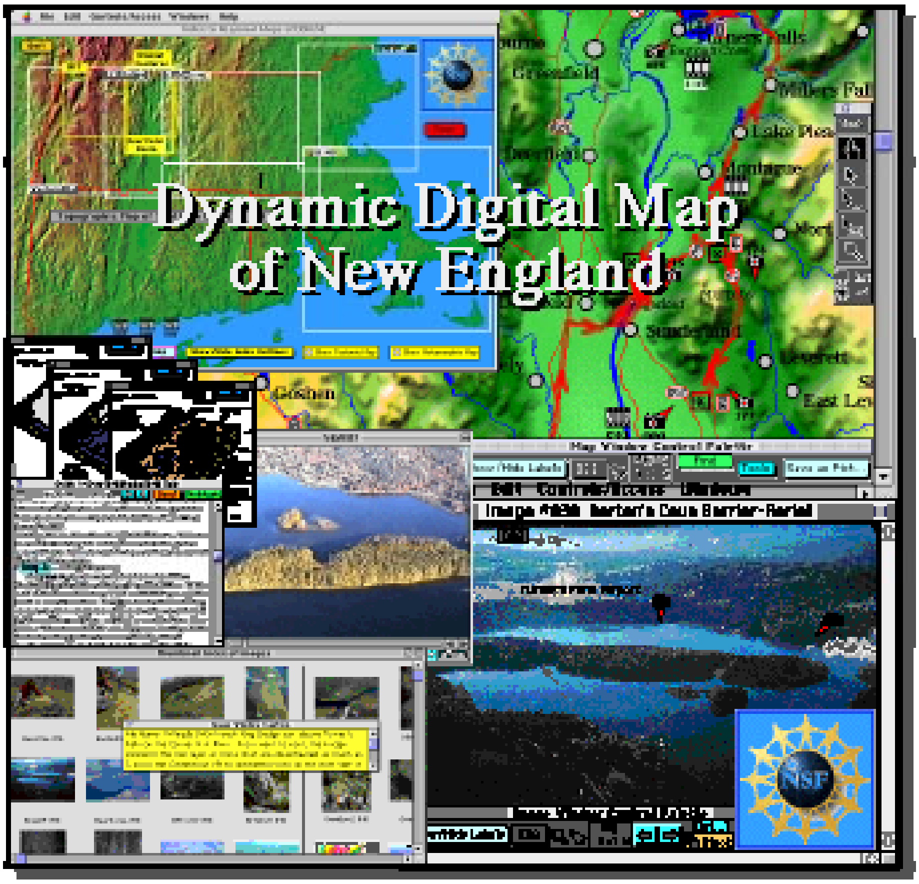
DYNAMIC DIGITAL MAPS
A MACINTOSH CD-ROM

BY CHRISTOPHER D. CONDIT



CONTRIBUTION NO. 72

DEPARTMENT OF GEOSCIENCES

UNIVERSITY OF MASSACHUSETTS

AMHERST MASSACHUSETTS

DYNAMIC DIGITAL MAPS
A MACINTOSH CD-ROM

By
Christopher D. Condit
Associate Professor

Contribution No. 72

Department of Geosciences

University of Massachusetts

Amherst, Massachusetts

April, 2000 with updates to 2002.01.07

ACKNOWLEDGMENTS

The work by Chris on the DDM Template and the DDM.TSP, which uses the DDM.Template as it's test-bed, was funded in part by grants from the National Science Foundation-DUE #9455563 and ILI #9750964 to Chris as the PI. Additional NSF funding by STEMTEC (DUE #9653966, a Collaborative for Excellence in Teacher Preparation) was paid to Chris during 1998. Chris also received Professional Development grants from the University of Massachusetts, and a grant from the Five Colleges Multimedia Access Project who funded a video camera. Contributions to the DDM.NE came in the form of a scan of the USGS Geologic Map of Massachusetts by Rotadyne, thanks to Dana Caron and permission to use the Color Shaded Relief Map of MA, CT and RI from Raven Maps and Images. Many students, as acknowledged in the individual DDM programs, contributed, some in the invaluable work-study program, others as interested volunteers or as a part of an independent study. The stand-alone project "Below Sea Level" was part of Paul Wernau's senior project - parts of it are included in modified form in DDMs. The Template was made much stronger by the comments of my students in my Fall 1999 class on how to make DDMs - we all worked hard in that class, had a lot of fun, and turned out some neat stuff. And Matt Neutra and I first taught each other this stuff back about five years ago, as part of his senior research project. Chris appreciates the reviews of Laurie Brown, his wife and a Professor of Geophysics in the department, and that of Rebecca Greenwood, who also contributed most of the work in compiling text and images for the all-day glacial field trip in the DDM.NE.

DDMs tend to be a synthesis of the work of many people, thus the compiler/programmer who puts these together should, in some cases, be thought of as more a glorified digital editor/programmer than as a scientific contributor. Despite this, the work of compiling a DDM can, and is, because of its inherent flexibility, an act of creation itself. This is especially true in the case of the DDM.NE (which drew upon the work of many of the geology faculty over the last decade) and the DDM.TSP, which drew on the work of two of our department faculty (Laurie Brown and Mike Rhodes) and was part of Andy Wulff's dissertation work, and was the initial brain-child of Mike Dungan at the University of Geneva. Brad Singer contributed heavily in the form of his map for that DDM. The Geological Society of America, who despite the Macintosh only restriction of the SuperCard programming environment, published the DDM.SVF on CD in 1995, gave me permission to continue getting the program out. They also provided some funding for on-going work on DDMs, and I am grateful for their support (especially that of Jim Clark, who was their copy right officer and much more), and for the help of Dave Schleicher, in his role as GSA's map editor/reviewer. The geologic map which formed the basis for the DDM.SVF was printed by the USGS late in 1999 as MI 2431.

AUTOMATED TOURS OF DDMS

Two of the Dynamic Digital Maps (DDMs) on this CD-ROM include completely automated tours of the DDM, which can be run by a single mouse-click on a "Tour" button found on their "home" or Index Map screens.. These tours show some of the features of, and how to use a DDM. Each lasts about six minutes. I built these tours so that you could painlessly decide whether DDMs are worth any further investment of your time, and to give you a jump-start into the very low-learning curve required to use DDMs. The tour on the Dynamic Digital Map of the Springerville Volcanic Field (DDM.SVF) emphasizes how to display different thematic maps and associated images, find samples, units and vents, and access a large analytical data set and an introductory text. The second tour, on the Dynamic Digital Map of New England is designed to show how to take one of the six field trips to western New England, and access the cross-linked guidebook article, view QuickTime movies and animated diagrams and find features on the maps. Each tour ends by showing you the DDM Overview window, where more details about the DDM can be found.

To run the tour on DDM.SVF, insert the CD-ROM into your drive, and open the CD-ROM entitled "CD_Make&UseDDMs". On the CD's window, find and double-click on the program* "DDM.SVF.v.2000.02.07". A splash screen will open; click on the "OK" button. An "Introduction to DDM-SVF" screen will next appear, and behind it in the upper left, you'll find an "Index Map" screen. Click on either of the yellow "Take a Tour" buttons on these windows, and then on the blue "Start Tour" button in the pop-up screen, and sit back and watch. Moving the mouse while "on tour" will interrupt the tour, with a message asking you if you want to quit. As noted in the first introductory window to the tour, you can quit or pause the tour at any time, and from this first "pop-up" window, jump quickly ahead to a step to resume a tour you have quit.

To run the tour on the Dynamic Digital Map of New England, insert the CD-ROM into your drive, and open the CD-ROM entitled "CD_DDM.NE". Drag the program* "DDM.NE.2000.04.10b" (or whatever date is appended to "DDM.NE") to your desktop. With the CD-ROM still in your drive, double-click on the program on your desktop, and then close the first splash-screen window. On the "Welcome" window that next opens, you'll find a red "Tour" button. Click on it, and then on the blue "Start Tour" button in the next pop-up window, and sit back and watch. Moving the mouse while "on tour" will interrupt the tour, with a message asking you if you want to quit. As in the DDM.SVF tour, you can quit or pause the tour at any time, and from this window, jump quickly ahead to a step to resume a tour you have quit.

This CD-ROM has been burned with three "volumes", so that three CD-ROM icons appear on your desktop when you insert the CD. To eject the CD-ROM, you must drag all three to the trash.

*Your CD may contain a later version of this program with a newer date - if so, use that one.

TABLE OF CONTENTS

Chapter Page

Chapter Page

Chapter 1
AN OVERVIEW OF THIS CD-ROM

Introduction

This CD-ROM contains programs, images and data used with and made by SuperCard, a high-level object-oriented graphical programming environment for the Macintosh computer. For details about SuperCard and DDMs see the section on the DDM.Tutorial and the DDM.Template below and the manuscripts contained in those directories on the CD. Because the programs on this CD are in a constant state of evolution, and many of them are works-in-progress, we plan to keep a limited number of these CDs in stock, and update them as changes are made to the various programs. See Appendix 1, a list of the files on the CD, where updated file information will be indicated. In many cases, where image or map content is not changed, these updates will be posted on Chris' web site where they can be downloaded and run with this CD (see the URL www.geo.umass.edu/faculty/ccondit.html.

Hardware and Software Requirements

The use of this CD requires a Macintosh computer with a CD, running Mac OS 7.5 (or above), and in the case of the stand-alone programs, from 10 to 20 MB of free RAM. All "stand-alone" (see below) programs can be run with no additonal software. Where noted below, some of the programs must be copied from the CD to your hard drive to run, and have the CD resident in its drive from which to access images and data. SuperCard v.3.6 is required to use the Tutorial and Templates to make your own DDMs. Except for the "ReadMe" files, which are stored as ASCII text files, the other texts files on this CD are saved in MS Word 5.1a format, and can be converted to other formats using MacLink Plus if needed.

Overview of Stand-Alone Programs and Data Folders

The files and folders on the CD have been written to CD as three "volumes", in order to streamline the data access for the progams DDM.NE and DDM.TSP, which display data from the respective volumes CD_DDM.NE and CD_DDM.TSP . To remove the CD from you CD-drive, you must drag all volumes to the trash. There are a total four independent "stand-alone" programs and three main folders, containing various manuscripts and SuperCard programs and SuperCard "shells" or "templates" on this CD. See Appendix 1 for a listing of the files and folders and their location on the volumes. The main programs and their volume are:

- •**DDM.NE** (Dynamic Digital Map of New England) (on the volume CD_DDM.NE)
- •**DDM.SVF** (Dynamic Digital Map of the Springerville Volcanic Field) (on the volume CD_Make&Use_DDMs)
- •**DDM.TSP** (Dynamic Digital Map of the Tatara-San Pedro Volcanic Complex) (on the volume CD_DDM.TSP)
- •**Below Sea Level**. (on the volume CD_Make&Use_DDMs)

You can run DDM.SVF and Below Sea Level directly from CD by simply double-clicking on their icons. To run both DDM.NE and DDM.TSP, you need to drag the program to your hard drive and run it from there, with the CD in the drive.

Description of the Stand-Alone Programs

DDM.NE contains maps, images, movies and text for six geologic field trips in western Massachusetts, and shaded topographic maps and the state geologic map for the entire state of Massachusetts. A topographic map for the Town of Leverett, where three of the field trips are located, which includes the linework for the bedrock geology of the western part, as compiled by Lisa Oxboel for her M.S. Thesis under Prof. Peter Robinson, is also included as a demonstration of future possibilities. The program includes a "tour" button for a completely automated guided tour of how to use the program. Just click it and let her run! It's my way of helping one get over the "I don't want to learn another piece of new software blues". See the below "Teaching Examples" heading for ideas on how to use (and we in the department use) this program in physical geology laboratory exercises.

DDM.SVF is a geologic map of the Springerville volcanic field in east-central Arizona. The program is an updated version of that published on CD by the Geological Society of America in 1995. It includes 74 pictures of the field, and lots of geochemical and geophysical data. It also includes a "tour" button for a guided tour of how to use the program. See the below "Teaching Examples" heading for ideas on how to I use this program in my junior level igneous petrology laboratory exercises. Included is a geochemical data set for use with Mike Carr's IgPet program (see the description in the SVFGeochem Folder below). Although the SuperCard programming is the most primative, in terms of content, this is the most complete of the DDMs on this CD.

DDM.TSP is a geologic map of the Tatara-San Pedro Volcanic Complex, located in south-central Chile. It contains a geologic map, modified from that published by Singer and others (1996) in the Geological Society of America Bulletin, along with 191 images and about 600 major and trace-element chemical analyses, 200 paleomagnetic site analyses, and 50 K-Ar and Ar age dates, most with their sites plotted on the images. This program represents the most recent application of the DDM.Template (see below) also found on this CD. As such, it should be used as a model by those interested in building their own SuperCard DDM. In terms of content, and links between the data sets as shown on the map and images, this is the most "in progress" of the DDMs on the CD, and will likely have updates in the near future.

Below Sea Level is a senior thesis done by Paul Wernau a few years ago, which I have kept updated with changes demanded by the evolution of the Mac OS. It is designed to be an interesting, fun and educational introduction to the geology of the ocean floor. The program is actually located in the folder "Below Sea Level", because it needs to point to the subfolder "QuickTime Movies" located there to play them. I have created a "pointer" or alias at the CD's "root" level so the program can be played by double clicking on it, instead of asking a user to open the folder with the program itself, and start it from there.

Description of Main Folder Content

These three main folders are found on the volume CD_Make&Use_DDMs. See Appendix 1 for a listing of the files and folders on the volumes.

The DDM_TutorialFolder

The DDM_TutorialFolder contains a manuscript (in MSWord 5.1) entitled "*A Tutorial to Using SuperCard v. 3.6 in Making Dynamic Digital Maps*" and the associated data files along with a SuperCard program "*DDM-Tutorial*" that I developed to familiarize interested parties with this approach to making dynamic digital maps with SuperCard. It should be a useful jumping off point for those who wish to evaluate the possibility of using this method of desktop publishing. The folder also contains a SuperCard program, "DDM_Tutorial.v.99.09.21c", to be used in the exercise described in the manuscript, and two small associated text files for use with it.

The DDM.TemplateFolder

The DDM.TemplateFolder contains a manual on how to use the contents of this folder to make a Dynamic Digital Map. It also contains the file structure (folders) for its use, with dummy data for illustrative purposes, and several ancillary SuperCard programs to be used in this effort. It is essentially an outgrowth of a semester-long course I taught during the Fall of 1999 at UMass on "Desktop Publishing of Geologic Maps". It clearly needs to be refined, but contains the core of what is needed to efficiently publish maps for use on Macintosh OS computers, using SuperCard v3.6 in combination with CAD, image processing and word processing programs. Major components include:

- "TemplateCookbook.200.xx.xx" which provides step-by-step procedures for making a Dynamic Digital Map; accompanying files for this effort are included in the folder, and their use described in the cookbook.
- "DDM.Template.2000.xx.xx" is a SuperCard "shell" for the user to modify into a unique DDM,
- Also contained in this folder are several SuperCard programs designed to help make DDM templates, as described in the cookbook.

The DDM.TeachingExamples Folder

The TeachingExamples Folder includes several exercises using DDM.NE, DDM.SVF and Mike Carr's IgPet (Petrologic Modeling) program, for use in a junior level Igneous Petrology course.

- "DDM.NE.DeerFldBasinVRGeo101lab" is an example exercise in using DDM.NE as a virtual field trip to the Deerfield Basin. We have used this exercise to replace our three-hour-long physical geology (100 level) course's field trip when we've had sleet and freezing rain and could not go on the trip. It could equally well be used by students to "pre-run" the trip, thereby maximizing what they get from such an excursion.

- "IntroToDDM.SVF_Petro-2000.02.16" is a step-by-step cookbook on how a geologist might use the DDM.SVF as a tool to begin modeling the petrogenesis of magmas of the volcanic field. I use it as a precursor to getting students involved in petrogenetic modeling, described in the next two manuscripts.
- "PetModelLab1-2000.xx.xx" and "PetModelLab2-2000.xx.xx" describe in a cookbook way how to use Mike Carr's IgPet and mixing programs in exercises designed to help students appreciate mass-balance and how that works with the fractional crystallization process in magma petrogenesis. The first PetModelLab1 document has an accompanying questions sheet, "PetModelLab1Quest2000.02.16" to be answered while doing the lab. The second "PetModelLab2" document asks the students to write a short research paper on their work, which involves selecting a parent-daughter pair of flows from anywhere in the volcanic field that might be related by fractional crystallization, and testing their hypotheses.
- The document "SVF-IgPetRock&MinSym.pict" is a pict file which displays a key to the symbols produced by IgPet when it plots the SVF data. Each whole rock sample will be plotted with a symbol which corresponds to one of the 12 lithologic flow types each flow has been assigned; minerals likewise will be plotted as one of the six mineral types found in the SVF IgPet files.
- The documents TSPM&TrcElChem2000.04.04.msw and TSP_IgPet2000.04.04a.roc, the latter to be used with IgPet more modeling urposes with the DDM.Tatara San Pedro.

SVFGeochem Folder (found in the TeachingExamples Folder)
The folder **SVFGeochem Folder** contains a suite of petrologic data from the Springerville volcanic field that is formatted for ***IgPet*** (see below) to used with the two manuscripts, PetModelLab1-2000.xx.xx.msw, PetModelLab2-2000.xx.xx.msw. In addition to the files described in those manuscripts, these include several that might be used by anyone wishing to work in the SVF. All data sets except the first below are in similar format, and can be combined with other data sets in the IgPet or mixing programs of Mike Carr. They include:

- SelSuitMajEl&DCPTrc2000.01.24a - a selected suite of DCP analyses by Jeanne Cooper Nesbitt from her dissertation work with W.K. Hart at Miami University, Ohio. This data must be used alone; it is not in a format compatible with being combined with other SVF data in the IgPet program, because it includes trace element data not contained in other data sets.
- SVFMinChem2000.02.08b.min. Includes 158 phenocryst analyses from 33 units in the Springerville volcanic field, mostly done by me at Univ. New Mexico.
- SVF_518MajChem2000.02.08.roc. A complete suite of all major-element whole rock analyses from the SVF. File names are sample IDs linked to unit IDs; the samples are listed by unit ID (which corresponds to geographic subdivision). Each analysis includes the age and area of the sample's unit.
- SVFMinRxSymbFile.txt. A file which will display a key to the symbols for the different lithologies and geochemistry analyses in all SVF IgPet files. To display these, the file must be loaded into IgPet, and an XY plot made of

SiO_2 vs. MgO

- "SVF_IndivRxChemsIgPet", a folder containing an analysis for each sample ii the SVF, to be loaded individually into ***IgPet***. For example, file "104L_Qsc5" contains a whole-rock analysis (sample 104L) from unit Qsc5.
- "SVFMinsByFlowIgPet", a folder containing single files that each include the mineral chemistry for all minerals in lava flow's sample. For example, file "431SM_Qmb6.min" contains all the phenocyrst analyses from unit Qmb6, including olivine, clinopyroxene, plagioclase and titanifeous magnetite.
- "SVFIndivMinChemsIgPet" a folder containing files that each include a single mineral analysis, to be added individually into IgPet. For example, file "116L-Ol_Qba" contains an olivine phenocyrst analysis from unit Qba.

Note that the user can modify the contents of these data files, and eliminate or add any number of analyses (each is sorted on a separate line) at will and save the modified data as a "text" file, as long as they keep the first line, which defines the data format for the IgPet and mixing programs.

Notes

A Note About IgPet:

- The petrologic modeling labs rely on the use of ***IgPet*** and ***mixing*** programs, which are petrologic plotting and modeling programs written by Mike Carr of Rutgers University. Mike tells me his Windows versions "are reliable and regularly maintained. The Mac version remains in Beta stage, usable but imperfect." For information, you can e-mail him at carr@rci.rutgers.edu, or write to him at Terra Softa, 155 Emerson Rd., Somerset, NJ 08873.

A Note About Future DDMs:

DDMs by their nature tend to be on-going works in progress. Their open structure leaves room to grow, thus I include in their name, a date, intended to be the "version" number. More recent versions of many of these programs may appear, and can be easily made available on web sites. An occasional scan of my web site might turn up updated version of these "works-in-progress".

I am presently (as of April, 2000) working on porting the DDM idea from the SuperCard programming environment to MetaCard, which is a similar, but cross-platform operating system X-Talk environment (including Win32, Macintosh, Linux and other UNIX variants). I hope, by virtue of this port, to create something that others will take advantage of as a medium of "encapsulated" publication. This type of publication includes the program with the media/data, so that even "little" folks with less funding can publish and distribute high-quality color maps and linked images and data in stand-alone packages that do not require high-cost software to use.

Selected References:

Condit, C.D., 1994, Dynamic Digital Maps: An alternative for distributing high-quality-data intensive maps for the academic community: *in* Sandberg, P., ed., Notes of GSA Continuing Education Short Course #7, 1994 Annual Geological Society of America Meeting, Seattle, p. 6.1-6.8.

Condit, C.D., 1994, DDM-Tutorial: A Prototype Tutorial to Using of SuperCard in Making Dynamic Digital Maps: *in* Sandberg, P., ed., Notes of GSA Continuing Education Short Course #7, 1994 Annual Geological Society of America Meeting, Seattle, p. 6.9-6.16.

Condit, C.D., 1995, Dynamic Digital Map: The Springerville Volcanic Field: Prototype color digital maps with ancillary data Boulder Colorado, Geol. Soc. Am. Digital Pub. Series DPSM01MC (CD-ROM for the Macintosh); v. 4.10.95 size: 36.7 megabytes.

Condit, C.D., 1995, DDM.SVF: A prototype Dynamic Digital Map of the Springerville volcanic field, Arizona, GSA Today, v.5, p. 69, 87-88.

Condit, C.D., 1999, Components of Dynamic Digital Maps, Computers & Geosciences, v. 25, p. 511-522.

Condit, C.D., and Acosta, A.V., 1993, The Generation of Raster-format Geologic Maps Using Digital Image-processing on the Macintosh Computer: A tutorial guide, U.S. Geological Survey Open File Report 93-526, includes 5 text files, 4 program files, 59 image files, 15 ASCII data files, and 18 CLUT files, 63 p.

Condit, C.D., and Barrett, J. M., 1996, Dynamic Digital Atlas of the Moon (DDA-Moon): A Stand-alone Program to Display Nomenclature and Public Outreach Collections, and Display and Make Measurements on Image Files, EOS, Trans. Am. Geophys. Union, v. 77, p. F447.

Condit, C.D., Crumpler, L.S., and Aubele, J.C., 1999, Lithologic, age-group, magnetopolarity and geochemical maps of the Springerville volcanic field, east-central Arizona, (1:100,000): U.S. Geol. Survey MI Map I-2431, 5 sheets.

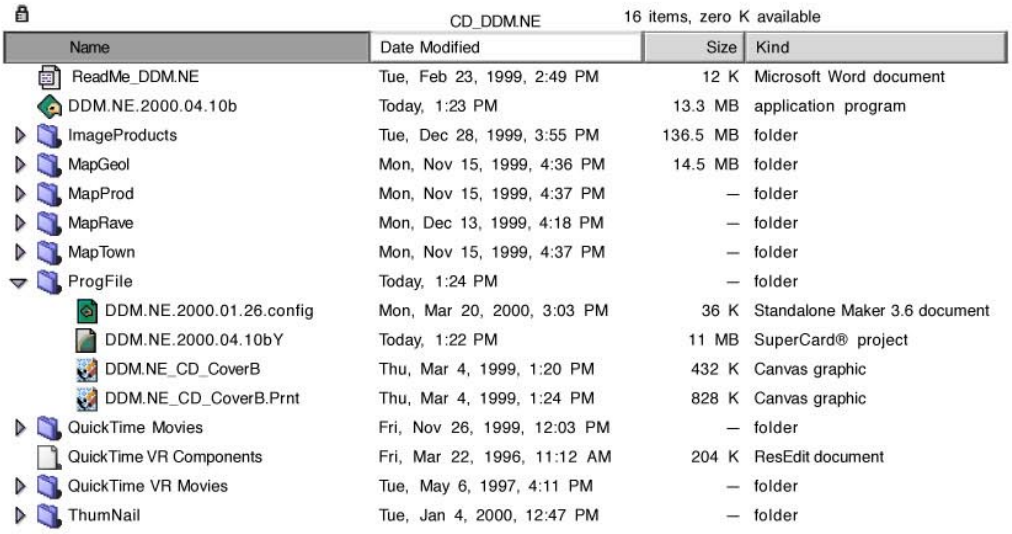
Condit, C.D., and Neutra, Matthew, 1992, SuperMap7/23/92: A prototype dynamic digital map, *in* Open File Report 92-416, compiled by Dave Traudt, Christine Cook, and Gregory Coats, Joint Education Initiative Sampler, CD ROM, 2.6 megabytes.

Condit, C.D., and Tanaka, K.L., 1995, Mars Geologic Landform Atlas and Reference Source (MGLARS): A stand-alone hypermedia program designed for geologic research, education and map/image/data presentation, Lunar and Planetary Science XXVI, p. 271-272.

Singer, B.S. Thompson, R.A., Dungan, M.A., Feeley, T.C., Nelson, S.T., Pickens, J.C., Brown, L.L., Wulff, A.W., Davidson, J.P., and Metzger, J., 1996, Volcanism and erosion during the past 930 k.y. at the Tatara-San Pedro Complex, Chilean Andes, GSA Bull, v. 109, p. 127-14.

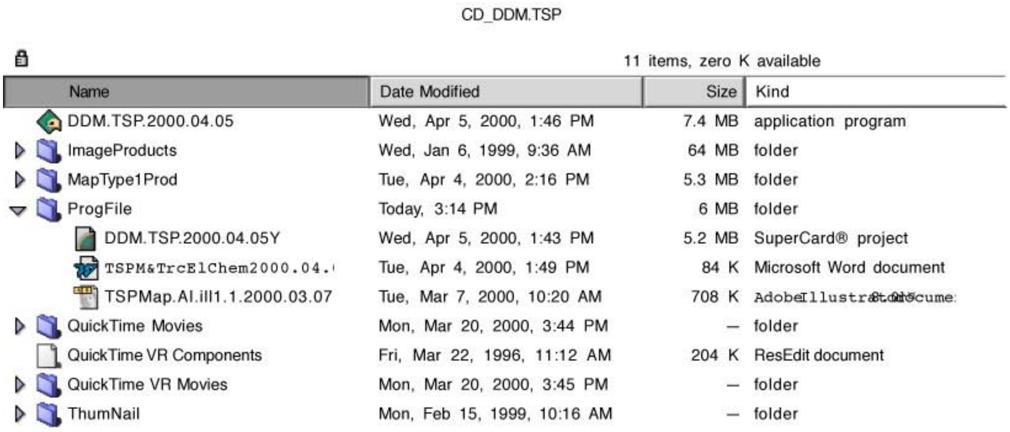
Appendix 1 - List of Directories and Files on CD

On the next pages, please find a list of the files and folders on the three volumes of this CD-ROM. Note that the names of many of these program files have a date attached (yyyy.mm.dd), which may not match those on these pages, indicating I have since updated that program with a newer version.



CD_DDM.NE 16 items, zero K available

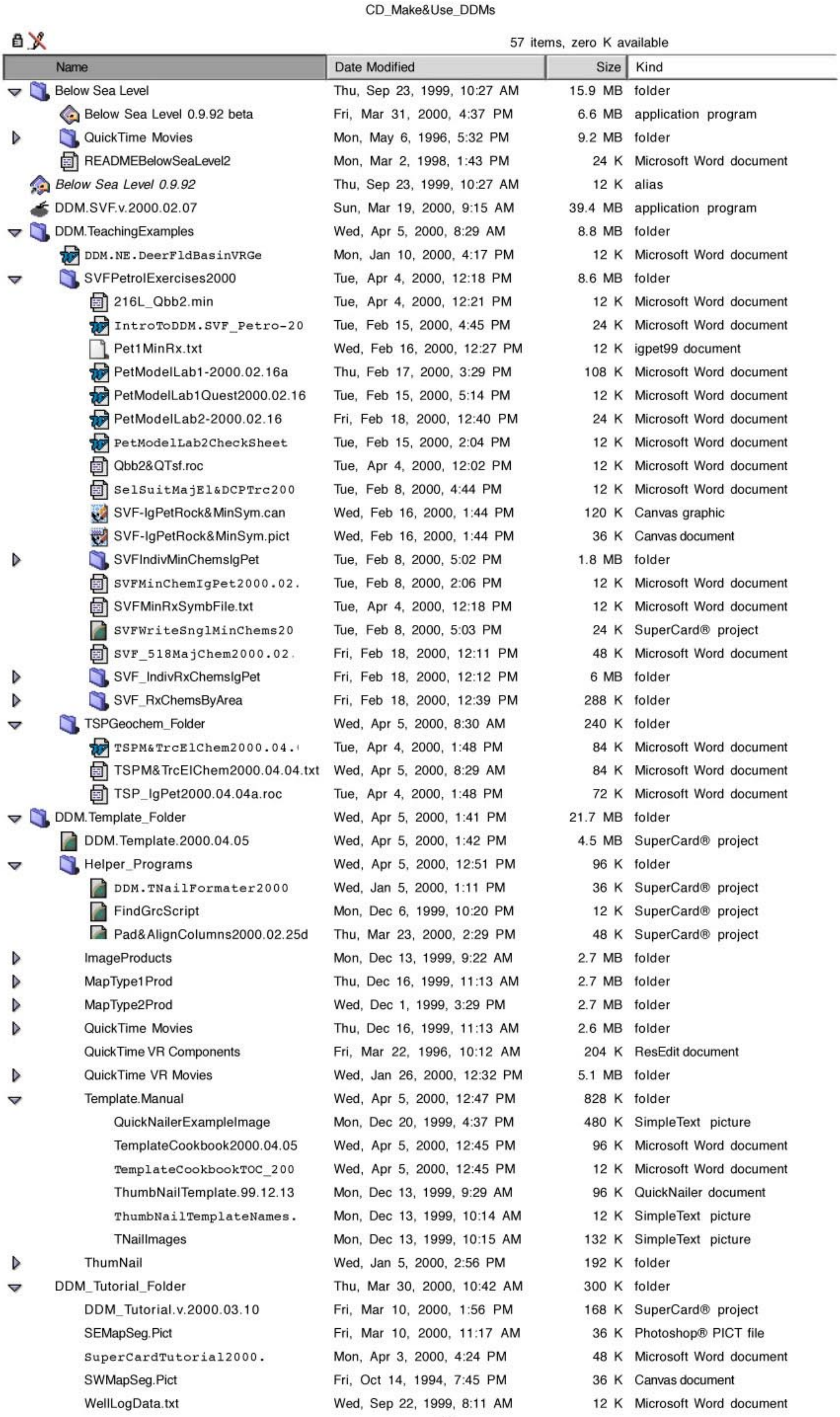
Name	Date Modified	Size	Kind
ReadMe_DDM.NE	Tue, Feb 23, 1999, 2:49 PM	12 K	Microsoft Word document
DDM.NE.2000.04.10b	Today, 1:23 PM	13.3 MB	application program
▷ ImageProducts	Tue, Dec 28, 1999, 3:55 PM	136.5 MB	folder
▷ MapGeol	Mon, Nov 15, 1999, 4:36 PM	14.5 MB	folder
▷ MapProd	Mon, Nov 15, 1999, 4:37 PM	–	folder
▷ MapRave	Mon, Dec 13, 1999, 4:18 PM	–	folder
▷ MapTown	Mon, Nov 15, 1999, 4:37 PM	–	folder
▽ ProgFile	Today, 1:24 PM	–	folder
DDM.NE.2000.01.26.config	Mon, Mar 20, 2000, 3:03 PM	36 K	Standalone Maker 3.6 document
DDM.NE.2000.04.10bY	Today, 1:22 PM	11 MB	SuperCard® project
DDM.NE_CD_CoverB	Thu, Mar 4, 1999, 1:20 PM	432 K	Canvas graphic
DDM.NE_CD_CoverB.Prnt	Thu, Mar 4, 1999, 1:24 PM	828 K	Canvas graphic
▷ QuickTime Movies	Fri, Nov 26, 1999, 12:03 PM	–	folder
QuickTime VR Components	Fri, Mar 22, 1996, 11:12 AM	204 K	ResEdit document
▷ QuickTime VR Movies	Tue, May 6, 1997, 4:11 PM	–	folder
▷ ThumNail	Tue, Jan 4, 2000, 12:47 PM	–	folder



CD_DDM.TSP

11 items, zero K available

Name	Date Modified	Size	Kind
DDM.TSP.2000.04.05	Wed, Apr 5, 2000, 1:46 PM	7.4 MB	application program
▷ ImageProducts	Wed, Jan 6, 1999, 9:36 AM	64 MB	folder
▷ MapType1Prod	Tue, Apr 4, 2000, 2:16 PM	5.3 MB	folder
▽ ProgFile	Today, 3:14 PM	6 MB	folder
DDM.TSP.2000.04.05Y	Wed, Apr 5, 2000, 1:43 PM	5.2 MB	SuperCard® project
TSPM&TrcElChem2000.04.	Tue, Apr 4, 2000, 1:49 PM	84 K	Microsoft Word document
TSPMap.AI.ill1.1.2000.03.07	Tue, Mar 7, 2000, 10:20 AM	708 K	AdobeIllustrat8.0®cume
▷ QuickTime Movies	Mon, Mar 20, 2000, 3:44 PM	–	folder
QuickTime VR Components	Fri, Mar 22, 1996, 11:12 AM	204 K	ResEdit document
▷ QuickTime VR Movies	Mon, Mar 20, 2000, 3:45 PM	–	folder
▷ ThumNail	Mon, Feb 15, 1999, 10:16 AM	–	folder



CD_Make&Use_DDMs

57 items, zero K available

Name	Date Modified	Size	Kind
▽ Below Sea Level	Thu, Sep 23, 1999, 10:27 AM	15.9 MB	folder
Below Sea Level 0.9.92 beta	Fri, Mar 31, 2000, 4:37 PM	6.6 MB	application program
▷ QuickTime Movies	Mon, May 6, 1996, 5:32 PM	9.2 MB	folder
READMEBelowSeaLevel2	Mon, Mar 2, 1998, 1:43 PM	24 K	Microsoft Word document
Below Sea Level 0.9.92	Thu, Sep 23, 1999, 10:27 AM	12 K	alias
DDM.SVF.v.2000.02.07	Sun, Mar 19, 2000, 9:15 AM	39.4 MB	application program
▽ DDM.TeachingExamples	Wed, Apr 5, 2000, 8:29 AM	8.8 MB	folder
DDM.NE.DeerFldBasinVRGe	Mon, Jan 10, 2000, 4:17 PM	12 K	Microsoft Word document
▽ SVFPetrolExercises2000	Tue, Apr 4, 2000, 12:18 PM	8.6 MB	folder
216L_Qbb2.min	Tue, Apr 4, 2000, 12:21 PM	12 K	Microsoft Word document
IntroToDDM.SVF_Petro-20	Tue, Feb 15, 2000, 4:45 PM	24 K	Microsoft Word document
Pet1MinRx.txt	Wed, Feb 16, 2000, 12:27 PM	12 K	igpet99 document
PetModelLab1-2000.02.16a	Thu, Feb 17, 2000, 3:29 PM	108 K	Microsoft Word document
PetModelLab1Quest2000.02.16	Tue, Feb 15, 2000, 5:14 PM	12 K	Microsoft Word document
PetModelLab2-2000.02.16	Fri, Feb 18, 2000, 12:40 PM	24 K	Microsoft Word document
PetModelLab2CheckSheet	Tue, Feb 15, 2000, 2:04 PM	12 K	Microsoft Word document
Qbb2&QTsf.roc	Tue, Apr 4, 2000, 12:02 PM	12 K	Microsoft Word document
SelSuitMajEl&DCPTrc200	Tue, Feb 8, 2000, 4:44 PM	12 K	Microsoft Word document
SVF-IgPetRock&MinSym.can	Wed, Feb 16, 2000, 1:44 PM	120 K	Canvas graphic
SVF-IgPetRock&MinSym.pict	Wed, Feb 16, 2000, 1:44 PM	36 K	Canvas document
▷ SVFIndivMinChemsIgPet	Tue, Feb 8, 2000, 5:02 PM	1.8 MB	folder
SVFMinChemIgPet2000.02.	Tue, Feb 8, 2000, 2:06 PM	12 K	Microsoft Word document
SVFMinRxSymbFile.txt	Tue, Apr 4, 2000, 12:18 PM	12 K	Microsoft Word document
SVFWriteSnglMinChems20	Tue, Feb 8, 2000, 5:03 PM	24 K	SuperCard® project
SVF_518MajChem2000.02.	Fri, Feb 18, 2000, 12:11 PM	48 K	Microsoft Word document
▷ SVF_IndivRxChemsIgPet	Fri, Feb 18, 2000, 12:12 PM	6 MB	folder
▷ SVF_RxChemsByArea	Fri, Feb 18, 2000, 12:39 PM	288 K	folder
▽ TSPGeochem_Folder	Wed, Apr 5, 2000, 8:30 AM	240 K	folder
TSPM&TrcElChem2000.04.	Tue, Apr 4, 2000, 1:48 PM	84 K	Microsoft Word document
TSPM&TrcElChem2000.04.04.txt	Wed, Apr 5, 2000, 8:29 AM	84 K	Microsoft Word document
TSP_IgPet2000.04.04a.roc	Tue, Apr 4, 2000, 1:48 PM	72 K	Microsoft Word document
▽ DDM.Template_Folder	Wed, Apr 5, 2000, 1:41 PM	21.7 MB	folder
DDM.Template.2000.04.05	Wed, Apr 5, 2000, 1:42 PM	4.5 MB	SuperCard® project
▽ Helper_Programs	Wed, Apr 5, 2000, 12:51 PM	96 K	folder
DDM.TNailFormater2000	Wed, Jan 5, 2000, 1:11 PM	36 K	SuperCard® project
FindGrcScript	Mon, Dec 6, 1999, 10:20 PM	12 K	SuperCard® project
Pad&AlignColumns2000.02.25d	Thu, Mar 23, 2000, 2:29 PM	48 K	SuperCard® project
▷ ImageProducts	Mon, Dec 13, 1999, 9:22 AM	2.7 MB	folder
▷ MapType1Prod	Thu, Dec 16, 1999, 11:13 AM	2.7 MB	folder
▷ MapType2Prod	Wed, Dec 1, 1999, 3:29 PM	2.7 MB	folder
▷ QuickTime Movies	Thu, Dec 16, 1999, 11:13 AM	2.6 MB	folder
QuickTime VR Components	Fri, Mar 22, 1996, 10:12 AM	204 K	ResEdit document
▷ QuickTime VR Movies	Wed, Jan 26, 2000, 12:32 PM	5.1 MB	folder
▽ Template.Manual	Wed, Apr 5, 2000, 12:47 PM	828 K	folder
QuickNailerExampleImage	Mon, Dec 20, 1999, 4:37 PM	480 K	SimpleText picture
TemplateCookbook2000.04.05	Wed, Apr 5, 2000, 12:45 PM	96 K	Microsoft Word document
TemplateCookbookTOC_200	Wed, Apr 5, 2000, 12:45 PM	12 K	Microsoft Word document
ThumbNailTemplate.99.12.13	Mon, Dec 13, 1999, 9:29 AM	96 K	QuickNailer document
ThumbNailTemplateNames.	Mon, Dec 13, 1999, 10:14 AM	12 K	SimpleText picture
TNailImages	Mon, Dec 13, 1999, 10:15 AM	132 K	SimpleText picture
▷ ThumNail	Wed, Jan 5, 2000, 2:56 PM	192 K	folder
▽ DDM_Tutorial_Folder	Thu, Mar 30, 2000, 10:42 AM	300 K	folder
DDM_Tutorial.v.2000.03.10	Fri, Mar 10, 2000, 1:56 PM	168 K	SuperCard® project
SEMapSeg.Pict	Fri, Mar 10, 2000, 11:17 AM	36 K	Photoshop® PICT file
SuperCardTutorial2000.	Mon, Apr 3, 2000, 4:24 PM	48 K	Microsoft Word document
SWMapSeg.Pict	Fri, Oct 14, 1994, 7:45 PM	36 K	Canvas document
WellLogData.txt	Wed, Sep 22, 1999, 8:11 AM	12 K	Microsoft Word document

Chapter 2

DDM-TUTORIAL
A TUTORIAL TO USING SUPERCARD V. 3.6
IN MAKING DYNAMIC DIGITAL MAPS

Introduction

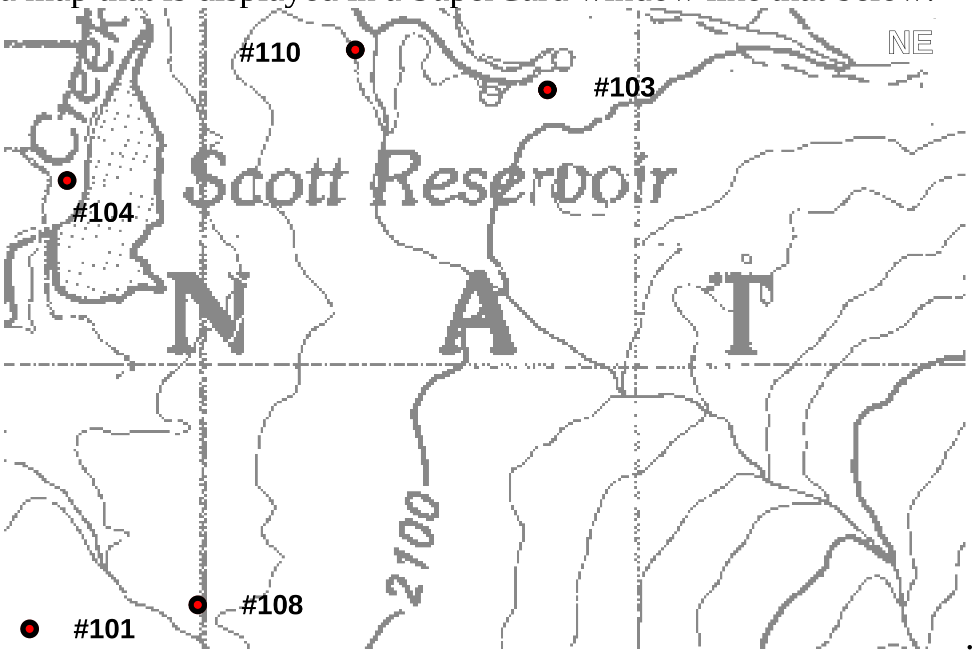
The purpose of this short article is to give you a "jump-start" into the use of SuperCard. I center the discussion around and introduce the parts of SuperCard that I have found to be most useful in making Dynamic Digital Maps (DDMs). This article is not intended to be at all comprehensive, but rather to give you a feel for what is involved in using SuperCard for one specific use. As covered in the reprint of the April 1995 GSA Today article (which can be accessed from the above URL), DDMs are not capable of making a map; that job is done outside of SuperCard, using standard programs such as word processors, spreadsheets and computer aided drafting programs (CADs). These maps and associated data are then inserted into a DDM which displays them, taking advantage of the computer environment and turning them into an interactive or "dynamic" map in the form of a stand-alone application. I key this discussion around a SuperCard project named "DDM-Tutorial.v.2000.xx.xx". To use this tutorial you will need SuperCard version 3.6; look for an updates of this tutorial at the above URL.

How I think of SuperCard

In my work I think of SuperCard as a *high-level graphics-based object-oriented* programming environment. *High-level*, meaning that you don't have to get down to command-line programming to, for example, define a window-SuperCard does that for you. *Graphics-based*, in that most of what I do with it involves pictures, figures and in the case of text and data, ways to display that graphically or put it into windows. *Object-oriented* in that virtually everything in SuperCard can be thought of as an "object" - that is a thing (like a coffee cup, a "thing") which you can go to, or show or hide, for example. These objects are special kinds of "things" in that they can be given information, attached to each as script (which can be thought of as hidden program lines) that will, if asked, do something. Objects include, to name a few, windows and cards (the SuperCard "framework"), images, labels on a map, figure captions, buttons, and containers. Containers, which are objects in which you can store values (numbers, text, and data), and are divided into two types: variables and fields

Objects - An Example of Inheritance

We'll talk later about how to get objects into SuperCard, or create them in SuperCard, but for now, to drive home the importance of objects, let's think about a map that is displayed in a SuperCard window like that below:



Included on this map are a large number of sample sites. You've decided it would be useful to see the data associated with each sample site when you click on it. Thus the function of each site on this "smart" SuperCard map is the same: to call data up from a table. One of the most useful things about these sample-site objects (or any SupeCard object) is that once you have created one, and attached to it a script to make it do something, you can duplicate it, and all the "attributes" you have given that object are duplicated with it. SuperCard calls the attributes *properties*, and they determine the way objects look and act. That means an object's color, size, shape, and attached script are included with the duplicate copy, making this "inheritance property" an especially powerful tool when dealing with large data sets associated with similar types of objects. In our example, you can create a script for a sample, and test that it does what you want. Once you're convinced it works, you can copy the sample, and to preserve each sample's unique identity (so it will call up the appropriate data) simply change the duplicate's name (a SuperCard property) and the graphics text (in SuperCard this is the textdata property) on the map (say from sample #100 to #101). The duplicated script carried with this sample will then do exactly what its parent object's script did. It will also inherit the parent's font, size, color, etc.

The SuperCard Programming Environment

The SuperCard programming environment is made up of three components: an Editor, SuperCard itself, used as a debugger, and a third that helps you make projects into programs that run independantly of SuperCard. The SuperCard Editor, known as SuperEdit, is used to build your project. Here you create your Windows, Cards and Menus, write your script and insert (or create) figures, data (and in this case maps). After you have created a workable project, you save it and hit "command-r" to get into SuperCard to run and interactively debug your project. After a project is created and is ready to be distributed, you use another program "StandAlone Maker" to convert the project into a stand-alone program.

The SuperCard Editor

Let's start SuperEdit, by locating it in the SuperCard Folder, and double-clicking on it. When you first start the SuperCard Editor, and load a project, the first window seen contains a list of all the names of the windows in the project. Let's open the DDA_Tutorial program by going to the "File" Menu, selecting "Open" and navigating to the folder where the project "DDA_Tutorial" is located, and double-clicking on it. Double-click on the name "Map" in this list, which opens another window that has a list of all the names of the cards in that window. Another double-click on the name of one of those cards opens the card. At this point you are in the *card-editing window*, and can work on the card. Let's make a new card, by first clicking back to the window containing the list of names of cards, and then pressing the "command-n" keys.

While in this new card let's look at the tools you use in the SuperEdit environment. There are four different types of *tools* available for working on cards, divided into four different tool palettes, including the *Draw, Paint, Field and Button* palettes. Any one of these palettes is displayed sequentially on the upper left side of the card-editor window. To switch between them you click on the top icon of the palette. To draw a box on the card, for example, you would click into the Draw palette, and click on the box tool, and use the mouse and cursor to draw your box. Go ahead and fiddle with them, and when you're done, close the new card window.

Debugging Using SuperCard

Using SuperCard you run the project you have created, debug your work, and modify script, if need be. Let's start SuperCard by going to the File Menu, and selecting the "Run" option (shortcut use: cmd-r (-r) keys). Click the OK button on the welcome screen when the program starts. You are now using SuperCard to debug the DDM_Tutorial" project.

Lets' explore two very useful tools while running this program in SuperCard. The most valuable single item in SuperCard is the "*trace*" capability, which lets you step through your project while running it, one step at a time, to check variables and see what each step does. You can also modify script while in the trace mode. To get into the trace mode, while running the project, press the "command-m" keys, which will bring up the "message window". Then, in the top line of that window, enter the word "trace" and hit the "return" key. The next action you start in the program will then be stepped through, one step at a time, with each object and its associated script and its variables displayed in the pop-up window. After you've type "trace" into the message window, and hit the 'return" key, click on the yellow "Trace Test" button and read the information that you see in the ScriptTracer Window. Be sure to click the "Go" button in the lower right of the ScriptTracer window before you do the next good stuff below.

Another extremely valuable tool is the SuperCard *Help* program, which is more-or-less a built-in SuperCard manual. You launch the Help program while running a project in SuperCard by typing the word "help" into the message window, and pressing the "return" key. When done using it, say, to look up an example of script for a command, you close the Help window to get back to running your project in SuperCard, or to edit the script within SuperEdit. Cruise through it a bit, and then close this window, and also the message palette.

Next we'll stop running this DDM_Tutorial in SuperCard and return to the SuperEdit. To do this, click on the "Quit" item in the "File" Menu. This shuts down SuperCard, and on returning you SuperEdit, leaves you with your "Project Window" open, which is where we want to be to examine the SuperCard framework a bit further.

The SuperCard Framework

Everything in a SuperCard project is viewed through a *window*. Windows come in various types and sizes, each of which you can define in the Editor or by using script. To see the different window types, select a window on the list of their names and press "command-i." Each window contains *cards*. Almost everything seen in a project is placed on a card. A window must contain at least one card. Cards can be of a different size than the window it belongs to; in the case in which a card is larger than its window, you would probably define that window as the kind you can scroll around in, so that you can view the entire card. Cards each have a *card layer*, and a *background layer*; the latter defines the card's size in lines and samples. Within a given window, cards may share a common background or have a unique background.

Another very important element in DDMs is the *field*, which are placed on cards while in the card-editor window, using the "field" tool in the SuperCard Editor. Fields are a type of *container*, and can be placed on a card, or in its background layer. Containers are simply a place to insert text or data in the case of DDMs, and can be either visible or invisible, two properties common to almost any object. These and Menus, found at the top of the screen in most programs, comprise the "framework" of SuperCard projects, and can be thought of as the "paper" on which maps are printed, albeit special paper, in that the paper itself can be "smart", because it can have script attached to it. Almost everything else in a DDM, I create outside of SuperCard itself, and paste into one of these framework components, while in the SuperEdit mode.

Script

Script is written in SuperTalk (ain't every thing "super" ? - wow, did I ever get panned by calling my first map "SuperMap"). An example of script: image a box on a card in a window, in which the words "Click Me" are written. Associated with this object (box) is some script. We'll get into how to write this script later, but the script looks like this:

```
on mouseDown
        answer "You clicked me!"
end mouseDown
```

Here's what happens when you click the box : "on mouseDown" sends the object a message that it has been clicked by the mouse button. The "answer" tells the program to make little "modal" window on the screen with the words "You clicked me!" in it. Because the answer command creates a modal style window, no other command is possible until the user clicks the "OK" button (or hits the return key). Try it while looking at the "Index Map" window of the Tutorial.

Navigation and Attaching Script to Objects

The tutorial is project in which I have already placed two of four overlapping maps (or map "segments"). These overlapping map segments are displayed one at a time in a "Map" window, on which there are four separate cards, one for each segment, named NWCard, NECard, SWCard, SECard. We will place one of the two additional segments into the map project in this tutorial, and examine the last card to see another way of displaying maps, other than permanently inserting them into projects. You cut the map into sections because you don't want any one map to be so big that a Mac with only 16 MB of RAM runs out of memory. Here we keep these segments small for use in the tutorial; in practice with 10 MB of free RAM, we can, with comfort, use 4 MB size map images (e.g an 8-bit 2000 x 2000 pixel size image). The map segments are opened or displayed by clicking on "buttons" in your project, found on a card (named "Index Map Card") in a different window (named the "Index Map" window). I design my DDM projects so that you always return to the "Index Map" window when a window containing a mag segment is closed. On this index map, you have a sketch of the entire map, and have also outlined the area of each map segment. Within each segment you have placed letters (here I call them NW, NE, SW, and SE); each of the 4 two-letter groups are discrete objects. You want the user to be able to click on one of these letters (for example, NE) and display the corresponding map segment. To get the program to do this, in the SuperEdit environment, you display the "Index Card". Next you select the letters "NE" (click on it once) and, from the Edit Menu of SuperEdit, select "Object Script". A window is opened, into which you can type your script. The script might look like this:

```
on  MouseUp
        open  card  "NECard"  of  window  "Map"
        close  window  "Index  Map"
end  mouseUp
```

When the mouse is released on object "NE", the program receives the mouseUp command, and looks in the object's script for the "*handler*" MouseUp. All "handlers" have the same basic structure, starting with an "on" , and terminated with "end." All commands within the handler are executed sequentially. In this case, the program sends an "open" command to the "Map" window; after it is open, it then opens the card named "NE Card." After this card is opened, the program closes the "Index Map" window. In this way the program navigates from one window to another. Anticipating that the user will want to return to the "Index Map" window when done viewing that map segment, the user might attach script to the "Map" window with similar instructions, telling the program "on closeWindow" to open the "Index Map" window, and then close the "Map" window. Try this in the Tutorial.

Inserting Maps into DDMs

Now, let's insert the third map segment into the "SWCard" The map has been created as a "Pict" file, in part using Canvas, and is saved on your disk as SWMapSeg.Pict." Double-click on the Window "Map" in your DDM-Tutorial" project, and when the list of cards in that window comes up, again double-click on the card "SWCard" to open it. Next, go to your "File" Menu, and slide down and release on the "Import Graphics" selection. Then, using the pop-up window, open the file "SWMapSeg.Pict." shortly, SuperCard will tell you that the graphics has been copied and is ready for pasting. Use the "command-v" keys to paste it. That map segment is in your project now. Note that the map segment has well sites on it, which were placed there using a CAD program (Canvas).

Inserting Map Segments into DDMs "On The Fly"

An alternative way to insert maps into a DDM is by creating a place on a card to temporarily put the map into when the user wants to display it, and removing it from the project when the user is done with it. This is done using the "pictureData" property of a rectangle. I have placed a rectangle named "theImageBox" on the SECard, and included the following script, attached to the card:

```
on  openCard
   put   "SEMapSeg.Pict"  into  ImageFileToOpen
   set  the  pictureData  of  cd  grc  "theImageBox"  to  ImageFileToOpen
end  openCard

on  closeCard
   set  the  picturedata  of  cd  grc  "theImageBox"  to  none
end  closeCard
```

When the card opens, this script looks for the image file named "SEMapSeg.Pict", which must be stored in the same directory (folder) as the program, and places into the rectangle using the the "pictureData" property. When the card is closed, the image is removed from the rectangle. This process is used instead of imbedding images into the program to keep program size to a minimum; among other reasons. Note that the locations of the wells are still located on the card "SECard". As explained below, they have script which calls up data. This attached script is a SuperCard attribute, which cannot be saved with an image file of the type that can be loaded as pictureData, thus we store these graphic objects, with their script, in the SuperCard program.

Fields and Variables

Now, let's look at the NW map segment and see how we can store data in a field, and tell a sample symbol when it is clicked on, to retrieve and display this data. First we have to create a field in which to store the data that we will associate with the map symbols. To make the field in the DDM, we'll go to the card containing the map segment in which the wells are located, and go to that card's background layer. To get to the background layer, use the "command-b" keys (or use the View Menu, and release on "Background"). Then, using the field tool, we click and drag out a field, and make it a scrolling type field. Next, with the newly created field selected, we'll name it. To do that, pull down the menu "Edit" and release on "Object-Info." In the pop-up window, we'll type in the name "WaterTableBedRxDepths". Later we'll look at the script I've attached to this field - here you're creating a similar field for practice! Our next job is to take a look at the data we'll be putting into this field. For simplicity's sake, we'll image a table of data for a series of wells; the data on each well is contained in one line. That line has the well number, the depth to water table, the depth to bedrock, and a comment like so:

#101	35	45	Jose logged
#102	37	50	Cailey logged
#103	45	66	Ian logged
#104	39	47	Laurie logged
#105	. . . .		

We have that data stored in a text file (called "WellData.MSWord"), created here, obviously using Microsoft Word. Go to that Word document, select the data and copy it to clipboard. Next go back to our card in the SuperCard project, and from the Field Tools palette,select the Text Editor (icon "A" on the palette). Click on the field and paste the clipboard contents into the field, using the "command-v" keys.

Next we want to link the well site symbol to the data field, such that when we click on it, the program will go to the data field, copy the correct line of data and display that on the screen. First we need to name the well site symbol on the map, just as we did in naming the field above. Now we have to provide script for this newly named object (say, well site "#101"), to link it to the data field. A simple way to do this is to again, select the object (#101) and using the Edit Menu, release on "Object-Script." In the pop-up window, we type the following script

```
on  mouseUp
        global  GlobWellToGet
        Put  short  name  of  target  into  GlobWellToGet
        send  showData  to  bg  fld  "WaterTableBedRxDepths"
end  mouseUp
```

The first line we use to declare a special type of container, called a *global variable*.. Any variable that will need to be recognized outside of the handler it is used in, must be declared a global variable. The next line tells the program, when the mouse is released on this object, to put its name (called the "*target*", here #101) into the global variable "GlobWellToGet." Next we tell the program to send a message to another handler named "showData" that we have programmed to be associated with the background field "WaterTableBedRxDepths."

The program now goes to the background of its card, finds the named field, and looks in that field's script for the handler "showData". It then executes the script within that handler. This script looks like this:

```
on  showData
        global  GlobWellToGet
        put  lineOffset  (GlobWellToGet,  bg  fld  WaterTableBedRxDepths,0)  into
lineTemp
        Get  line  lineTemp  to  lineTemp  of  bg  fld  WaterTableBedRxDepths
        put  it  into  thisWellsData

        if  lineTemp  =  0  then
                put  "Depths  for  "  &  quote  &  GlobWellToGet  &  quote  &  "  not  found"
into¬                   NoAnaylsisFound
                answer   NoAnaylsisFound
                exit  showData
        end  if

        answer   thisWellsData

end  showData
```

(Note: this little symbol in quotes "¬" which is seen at the end of the line that starts - Put "Depths for " & quote & GlobWellToGet & ...." means this line continues below.)

In this script we again declare the global variable "GlobWellToGet", so that within this handler, we can pull the word "#101" out of that container. The next line of script finds the line number within the data field that has the word "#101", and puts that line number into the (local) variable "lineTemp" (we use the global variable "GlobWellToGet" as if it were the word "#101"). The line of script starting with "Get line. . ." copies everything in this line (which is from line 1 to line 1, where 1 is the local variable lineTemp.) into the variable "it", which the program next puts into the variable "thisWellsData." Then the program checks to see that it found data. If it didn't, the variable "lineTemp" would be zero (0). IN this case, having found data, the program skips the "if. . . then" sequence of script. The last thing it does is "answers". The answer command tells SuperCard to put a "modal" window in the center of the screen, and in it put the words, in this case contained in the variable "thisWellsData." A bit fancier way to do all this can be found by exploring the script in the NE map segment, but that is beyond the script for this brief introduction.

A Fast Way to Name & Link Map Symbols

One last note: remember that "SWMapSegment" we inserted, and the well sites I mentioned? Because they are each separate objects, you can, using the Draw "text" tool, copy their name (highlight them with this tool, and use command-c), and then, by switching to the arrow (or pointer) tool select them and then use "command-i" to get information on them and insert (using "command-v") their name. All that's left to do with them is attach script to each, and they're ready to call-up data. To do that, copy the script from a well site in the NECard. Then, holding the command key down, select all the well sites on this SWCard. Once they're all selected, press the "command-e" keys to bring up all the selected objects' script windows. Then alternate the "command-v" (for paste) with the "command-w" (for close script window) keys and punch in the script for all sites. Note that in my DDM.Template program (in progress as of 9/97), I supply a tool that automates this "name & link" process. This tool copies the textdata of each object (of "text" type) and assigns it a SuperCard name that is identical. It also attaches appropriate script to each.

Summary

I hope from this you can see that with a little familiarization, programming with SuperCard is a reasonably straightforward process. More importantly, I hope you can begin to see the many ways this powerful graphics-based object-oriented programming environment can be used for geologic applications. This little tutorial and its associated program are obviously in the "beta" stage yet, but with a little work I hope to flesh it out, and make it more comprehensive to the point that the beginner has the ability to take it from there, and use all the power that SuperCards offers as a research and educational tool. I will be posting future updates of this on our anonymous FTP site at UMass at eclogite.geo.umass.ed (128.119.45.20) in the directory ccondit. It and several other SuperCard projects can be also obtained from the web page at URL:www.geo.umass.edu/faculty/ccondit.html.

Chapter 3

DYNAMIC DIGITAL MAP TEMPLATE COOKBOOK

Introduction to the Cookbook

This "cookbook" provides step-by-step procedures for making a Dynamic Digital Map. The process is done by modifying the SuperCard project "Dynamic Digital Map Template" (DDM.Template) and requires a basic familiarity with SuperEdit and SuperCard, and with the use of a CAD program, and a photo editing program such as Adobe Illustrator and Photoshop and a word-processing program such as MS Word. Reference is made to the use of Quicknailer and several ancillary SuperCard programs as well as the DDM.Template program itself, which are used in this process. The cookbook is an outgrowth of "crib sheets" created to help students taking my "Desktop Publishing of Geologic Maps" course in the Fall of 1999 at UMass.

The following are the major components that need to be modified to create your own DDM from the Template. These include:

- The "Index Map", the project's "home screen", which provides links to viewing other more detailed maps of the region.
- Overlay cards containing graphics and scripted objects for each of the detailed maps (or "map segments") of the region. These maps may be repetitive thematic variations of identical areas (e.g. a topographic map, and a photo-mosaic of corresponding and co-registered dimensions). The overlay may include only the two essential objects already included in the template, or may also include vector graphics, text, etc., which can be made visible or invisible as it overlays the maps (or images, see next bullet) displayed by the DDM program.
- Overlay cards containing graphics and scripted objects for each of the photographs or figures (graphs, tables, sketches) of features described in the text of the DDM. These overlay cards are similar to those discussed above for the maps.
- Text describing the map and or field trips or a series of field trips. A separate textural component is captions that describe each of the photographs/figures, and can be displayed by the user as each of the nine "image windows" is viewed.
- A "Thumbnail Index" of images used in the program, created from output of the program "Quicknailer" and a SuperCard ancillary program named "ThumbnailMaker".
- Three clicklists which are lists of each of the included a) maps, b) images and c) text articles in the project. These clicklists provide an index to the project, and access to each of the included components (maps, images, and guidebook (text) articles).
- This Template can also be used to include analytical data;

Note that this work is aimed toward finally burning your program and data onto a CD, which is a locked disk, to which you cannot write. Because the DDM.Template program modifies itself as it runs, the user cannot run it directly from the CD, but must first copy the program from the CD to a hard drive

(HD). If the CD is kept in the CD reader when the program is run from the HD, it can then access its data from the CD, without having to copy the large files (images, maps, and movies) to the HD. I recommend that you name programs with a period, or "." before the date and keep a period or "." after the letters "DDM" in the name. Thus for example, a program named DDM.NE.2000.01.25, results in the program assuming it has access to data on a CD named CD_DDM.SVF. I suggest that when you burn your CD, you should name it accordingly.

STEPS TO ADDING MAPS TO DDM.TEMPLATE

This assumes you have already created a vector (label) overlay for your Map. The program will place this overlay on top of your Map when it is displayed in the DDM; the overlay can be turned visible and invisible using the show/hide button in the program. The label overlay might consist of sample sites or graphic elements like lines, arrows, patterns and/or text etc.

Creating a Map Overlay

To create an overlay in Illustrator:

1) Start Illustrator
2) Open a "New" file
3) Use the "Place" command to insert your map on the background layer
4) Make a new layer in Illustrator in which to draw your overlay.
5) Place your labels/patterns, text, etc., taking care to place a short vertical line, starting at 0,0 in the upper left hand corner of the image, to be used or alignment purposes by the program (that is to line up the overlay with the image when displayed by the program). Use the "pen" tool with the shift key down to draw straight lines (click twice, once in upper left, then below it about 10 pixels or 1/4 inch).
6) When done adding labels, we next need to save the labels layer in a format that SuperEdit can import. To do this, first delete your bitmapped image. Then, use the File-Export menu to save the file in a Pict format, and save the remaining file with the same name as your image file, except, instead of using the three letter extension "MAP", use "LBL" (this just to keep your self organized - you might also want to save these in a separate folder).
7) Quit Illustrator.
8) Note Windows graphics from other PC-based CAD programs, which are saved in a "metaWindows" format can be converted to a pict file using MacLink Plus, and then imported into SuperEdit.

Parts of the Template to Modify to Add Your Map

Components that need modifying in the program include:

(a) A scrolling field in the window "Clicklist of Maps" (Wd ID 106)
(b) Adding a card with the same name as the map segment to the "Map Window Overlays" (window Wd ID 107)
(c) Modifying the Index Map to show the outline of each map segment, and allow the user to click on those outlines to open the map segments.
(d) Adding Camera Icons to Map Overlays, done while running the program in SuperCard.
(e) Adding a scale factor for each map segment, so one can use the Tools

Palette to make linear and area measurements. This is done while viewing the new map segment in SuperCard, by following the directions (below), and using a button on the Project Modifier page, accessed from the File Menu.

(f) Setting up the correct buttons, if needed, to display different types of thematic maps of the same area. These modification are made on the "Preferences and Set Up" window, ID 146).

In addition to modifying the program, you must be sure a copy of the actual map file has been placed into the correct folder, which should reside in the same directory as the program. In the DDM.Template folder I have created two folders for these map files, which might for example, include a scan of a topographic map and a correlative airphoto for each of the map segments, thus two different thematic types of maps of the same area(s). These are the two map folders "MapType1Prod" and "MapType2Prod". Each MUST contain the file "Dummy.MAP" and you should include copies of your map segment(s) in their respective folder. Since we assume the labels are identical for all co-registered thematic map segments, below we discuss adding only one map segment and its labels. We also discuss how to add geographic locations to the map overlay, a process done in the SuperEdit environment. Finally we describe how to modify the buttons to change map types, found in the "Preferences & Set Up" palette.

To modify the program:

1) Start SuperEdit

2) Open the project DDM.Template.2000.xx.xx

3) Save the project with another name (I suggest with the current date or a lower case letter added to the date, if you've already saved a version of it using today's date).

Adding the map's name to the "Clicklist of Maps"

We'll do this so the program can call it up from disk. We assume you have already saved the map image out of Photoshop into the folder "MapType1Prod". If not, please do this NOW. (Ideal map parameters are to save it as an 8-bit file, in a "Pict" format, with a size of about 1800 x 1600 pixels). Note that you can get away with a much larger file if you save it as a "bitmap" or black & white file (I do this and include in the overlay for this image a vector rectangle as a transparent overlay. When in SuperCard I "Get Info" on the overlay rectangle, and using the "Ink" button set the "XOr" off the vector rectangle to true, having already set the color of the rectangle to brown).

4) Double-click on (to open) Wd ID 106 (Wd # 7), and then double-click on the card (Cd ID 101 of Wd ID 106) to open it.

5) Click three times in the card's "Tool Type" selection box (upper left hand side) to select the "Field" Tools, then click on the letter "A", to select the "Text" tool within the field tools.

6) We need to make a line next with the data for your new image. either (a) find a line you wish to replace, scrolling down the field if needed, or (b) if there are none to replace, go to the next-to-the-last line on the list. The easy way to <u>add</u> a line is to copy the line above your line, and paste it back into

the field, and then modify this line. Note that the last line MUST start with M#zzz, or you won't be able to see the last line of "real" data so you have to insert new "last" lines above this M#zzz line. Do either (a) or (b) above, and then make sure the line you want to modify is visible in your field. NOTE also that the field trip or text discussion (one per map) associated with this map must have a three letter identifier (in the column TripID), e.g. FT1). This same identifier must be included in the ClickList of Field Trip Guides (window ID 108). This is essential so that the hyperlinks from map to text works correctly.

7) In the Finder, go to the Map file you wish to add to your project, click on it once, and copy the name. Note the last three letter of it's name MUST be "Map", either upper or lower case.

8) Go to the line you're modifying, and select the name of the file you are replacing, and use "cmd-v" to insert the new name. If needed, be sure to change the "M#" of the file to reflect adding a new image. Each image must have its own unique "M#". NOTE that the M# must correspond to the card number (NOT ID) below.

9) Add a key word identifying phrase that will help the user locate this file using the "find" or search capability in this window (which is more properly a palette). Remember this phrase is also attached to the window name when the image is displayed by the program.

10) Use "cmd-K' to both compact and save the program (do this "early & often" to preserve your work.

Inserting Map Window Overlays

Next, in the "Map Window Overlays" file, we'll make a new card (or modify an old one) and insert the label overlay you've saved out of Illustrator into this card.

1) Next, double-click on (to open) Wd ID 114 (Wd # 15). We assume you will be modifying an already existing card here. If one does not exist, select the last one in the window, and use "cmd-c" to copy it. Then, click on the left side of the list of cards, just below the last card, to set insertion point for this new card, and use "cmd-v" to paste the copied card. * Note that each card has a graphic named "ImageNumber" near the upper left side. The number in this

2) Now, check that the card you are modifying has a <u>Card Number</u> that matches the "M#" that you have assigned you new image in the "Clicklist of Maps" field. (The Card Number is the left-most number in the list of cards in the Map Window Overlays, NOT Card ID, which is the number in the second column). Also, note M# must have three digits, thus card 1 corresponds to an "M#" of M#001).

3) Then double-click on the wanted card in the window "Image Window Labels" to open it. Use the "View-Show Invisibles" menu to show any invisible labels on this card.

4) Use "cmd-a" to select everything. Then use "cmd-u" to ungroup these objects. Now, we need to remove everything on this card except two objects in the upper-left. We do this by holding the shift key down, and dragging the mouse over both the "ImageNumber" label in the upper left

side of the card, and the line which is in the extreme upper-left side of the card. This de-selects these two objects, leaving the others selected. Then we press the "Delete" key to remove the still selected objects.

5) To insert your labels, use the "File-Import Graphics" Menu. You'll get a note saying the graphics objects have been imported.

6) Use "cmd-v" to paste the label overlay into the card. If asked to add a "color-look-up table" say "No".

7) You may find letters that need changing, with font sizes that don't match, and colors that need to be modified. I often add a dark color rectangle to the card temporarily to aid in this work. After drawing one, with the object selected, use the cmd and minus key combination (that is, cmd -) to send it behind the labels in your overly, and don't forget to delete it when you're done with it. If you want, you can also import the image to the card (and then subsequently deleted it) to see if your colors for the label are OK. If you do this, you might want to temporarily add the "CLUT" to the project and then delete it. Yu do this by answering yes to the pop-up window that asks if you want to insert a new CLUT, and give it a name you'll remember. To delete it, goto the resources list in the project overview window, select the CLUT on that list and hit the delete key.

8) If you wish to have a field trip route included, my suggestion is to have that you draw it in Illustrator or Canvas, and group it, and then once it is imported into your map window overlay card, to name it "TripRoute (see the Template's first card in the window "Map Window Overlays" for an example). I addition, if you have numbered field trip stops, I recommend you copy the ones in that window (which are small white rectangular text objects), and use them, and if you need to add more, duplicate the ones in that window. However you create them, be sure to change the text label (in SuperCardese called the textdata) to correspond with that of the object's name. Remember to change the label, use the Vector Text Tool (select the letter "A" in the "Caliper" Tools), and click on the text to select and change it. To change the name, select the object, use the key combination "cmd-i", and in the pop-up window, enter your desired name. If you use the names "Stop 1 , Stop 2, etc., the Map Search Clicklist will properly recognize it, and sort it by object type, and it will be properly recognized in the hyperlink routines invoked from click and double clicks in the Map Text windows.

9) To change colors of the text, select the "vector" tools option on the left side of the card, and the "Text" tool (letter "A", and select the text whose color you want to change. Go to the lower left of the window, and click on the left-most box and then on the left-pop-up drag the mouse to the color you want, and release.

10) Next, look at the upper left-hand card graphic "ImageNumber", and be sure it matches that of your card number. If not, select the "Vector Tools" associated with your open card (the upper left hand box, should be a pair of calipers), and blow it select the letter "A" to edit vector text objects. Click in the ImageNumber graphic and change the number to match the card number & "M#". Note that it must have three digits, thus for example, card 2 should correspond to S#002", and the ImageNumber label should be "002".

11) Lastly, after removing unwanted labels, and modifying everything to your happiness, you need to group your new additions to the card. To do this, use the key combinations "cmd-A" (select-all) followed by "cmd-G" (to group the selected objects) and then close the card.

12) Now use "cmd-k" to save/compact your project, and then "cmd-r" to run the program in SuperCard to check your work. Call-up your new image from the Control/Access - Open/Close List of Images menu, and double-click on the image name in the list.

Adding Map Location Graphics to the Template

Location graphics are simply rectangles that you add into an overlay file, to which you add the name of a town, geographic feature or location (that is use the "name" property of SuperCard). The DDM.Template program can then be asked to search for and center the map on these graphic features, which are kept as invisible objects so as not to clutter the map.

1) Open one of the Map Overlay cards.

2) If you wish to see the overlay's associated map, I recommend you import it into the card's background (and when done with it, delete it). Do this by:
   a) selecting the background by using "cmd-b".
   b) use the "File-Import Graphics" Menu, and select the map file associated with this map overlay (it should be in you MapType1Prod folder).
   c) use "cmd-v" to paste the graphic object into the background, and be sure it is lined up with the upper left-corner of the map in the upper-left corner of the background (and card).
   d) use "cmd-b" to toggle back from the background layer to the card layer.

3) Select the "View-Show Invisibles" Menu

4) Select the Rectangle Tool in the Vector Tool Palette, and draw a rectangle at the location of your choice.

5) With this object still selected, use the "cmd-i" keys to "Get Info" on the object

6) Into the pop-up screen, enter the name you wish the object to have.

7) Under the "Graphic Properties, on the left side, be sure to uncheck (deselect) the "Visible" property.

8) To make additional named graphic objects, just duplicate this one (select it, and use the "cmd-d" keys). Then move it to the new location, if necessary, resize it, and use "cmd-i" to call-up the "Info" window so you can change the new object's name. Note I recommend you duplicate the first one, because it is essential that these objects are invisible, and thus have the "visibility" property set to "false" (that is unchecked).

9) When you're done, toggle to the background layer (see step 2a above), select the map and delete it, and then toggle back.

10) Now you need to group all the graphics object on the card layer, being sure to turn on the "invisible" objects when you group them. Use the "View-Show Invisibles" Menu to do this before you use "cmd-a" to select all, and "cmd-g" to group the objects.

Adding Scale Factors to Map Segments

Each new map segment should have a scale factor added so one can use the Tools Palette to make linear and area measurements. This is done while viewing the new map segment in SuperCard, by using a button on the Project Modifier page, accessed from the File Menu. Use of this button creates an invisible graphic, added to the mag segment's overlay that contains the number of km/pixel (and mi/pixel), which is easier thought of as the inverse of the number of pixels per km or mi (e.g. 1/pixel/km). If you already know the number of pixels in each km or mi, the set-up button gives you the option to enter that directly, and thus set your map segment's scale. If you don't know that, but do have a graphic showing some known distance, you can use that with the set-up button, and enter the know distance [either in km or mi - you can use fractions (e.g. 0.1 km for 100m], and draw a line of that distance to set your scale factor.

1) With your project in SuperEdit, open the Project Modifier window, from the "File" Menu.

2) Click on the button near the lower left of the window entitled "Set This Map Segment's Scale".

3) To the question "Do you want to set the scale for this Map Segment by Drawing a line on a scale included on your Map, or by Entering a Number?", answer by clicking either the "Draw" or "Enter Number" button. The ""Enter Number" Option is in (4), and the Draw Option is in (5) below.

4) If you choose the "Enter Number" option, then:

a) You will be asked to "Choose the units for which you will be entering the number of pixels", with the choice being either km or mi. Pick you units, or "Cancel" out, by selecting that option.

b) Next, you'll be asked to "Please enter the number of pixels/km" (or mi, if those are your units). Enter in a number, or select "Cancel" to bail out.

c) If you entered a unit, you'll get the notice that "Your scale for THIS map segment has been entered both for pixels/km and pixels/mile. Now I'll save your change!".

5) If you choose the "Draw" option, then:

a) You'll get the message "What units are you using on your map scale?" with two choices: "km" or "mi". Select one.

b) Then you'll be asked to "Please enter how many units (km or mi) long the line you going to draw is" Along with the message to "then go to your map window, select it, and finally, draw your line."

c) After you draw your line, you'll get the message "It looks to me that your line is x number of pixels/unit (mi or km). Is this OK, or do you want to try again?" and be given two choices: "Try Again" or "OK"

d) If you try again, you go through all steps (2 to here) again, if not you'll get a message that your changed scale is going to be saved into your DDM.

6) That's it. Your new map segment now has a scale. Note that if you don't know your scale (in pixels per km or mi), and choose to draw a line to get it in your first segment, you'll be given the number of pixels per unit at that point. If the rest of your map segments have similar scales, you can, there after enter the scale data for new map segments using the "Enter Number" option, which is faster.

Adding Map Access from the Index Map Window

1) Open wd id 103 (the Index Maps) window, and then open it's first card

2) If you haven't yet inserted your Index Map, delete the one in the Template program, and import your graphic. After you import it, select it (with a single click) and send your index map graphic behind all other objects in this card by using a "cmd -" key combination (that is, using the command and minus keys).

3) Go to the View and select the Show Invisibles.

There are two options to choose from next: A uses the green outlined graphics, and option B uses the Red outlined graphics objects. Those described in "A" (green) have the area to be shown outlined in objects that are normally invisible, but which become visible when you mouse over the associated label (e.g. green outlined NW)). The B option (outlined in Red) has no invisible objects, but is made of two components, one an outline the other a text object. To alter this one you have to "ungroup" the two, which will loose the original grouped object's text, so I recommend you "cmd-d" to duplicate it first and then alter it as per your choice below.

OPTION A

This assumes you will be using the green outlined objects, so select and delete the two red-outlined objects.

4A) Click & drag the outlines you see to correspond to the outline of your new map segment.

5A) Click on the segment label (e.g. NW, NE, SW, SE) and use "cmd-e" to open the object's script window.

6A) Replace the name "M#" to correspond to the one for your map segment.

7A) Change the text in the YellowInfo line to correspond to what you want the -pop-up info to say.

8A) Note: you may want to change the label or text for this graphic to something that better corresponds to your map (e.g. "NW" may change to "Central" or whatever.....

9A) Remove or add as needed the other graphics used to call-up maps from this Index Map window.

OPTION B

This assumes you will be using the red outlined objects, so select and delete the eight green-outlined objects.

4B) There are two red example objects to choose from; I recommend that you select one and use "cmd-d" to duplicate it and then work with the duplicate object.

5B) If you work with a duplicate of the "NW" object, for example, select it, and use "cmd-u" to ungroup it. Note that this loses the grouped object's script, so we'll get (copy) that from the original object when we get done altering this one, and attach the script to this one later. To do this, select the origianal "NW" object, use cmd-e to open its script, then in the script window, cmd-a to select all the script, and command-c to copy it. then close that window ond open the script window of you new object (slect it, and use cmd-e). then into that window use cmd-v to paste, and close it with a cmd-w.

6B) Move and alter the outline box to better delineate the area you will be calling up, and then move the lettering to a part of the polygon that doesn't hide the underlying index map. You may want a more irregular outline, in which case you might choose to draw you own, or use the other (Area SE) example's polygon. If you draw your own, use the appropriate draw tool, and make it's interior visible by setting the interior of it to "N", using it's setting selector in the lower left hand side of the window.

7B) Alter the text label to your liking (use the letter "A" in the vector tools), and then group the two objects (the outline and text) by selecting both and using "cmd-g".

8B) Select the original object and use "cmd-e" to open it's attached script. Use "cmd-a" to select the script and "cmd-c" to copy it. then close the script window.

9B) Select your newly made grouped object, and use "cmd-i" to get info on it, and type in a name of your choice for the new graphic. Then select the "script" button in this pop-up window.

10B) Paste the script you copied into this object's script window.

11B) Replace the name "M#" to correspond to the one for your map segment.

12B) Change the text in the YellowInfo line to correspond to what you want the Yellow pop-up info to say.

13B) Continue making as many outlines as you need, and delete those you don't use.

Setting Buttons to Change Thematic Map Types

Except for adding camera icons to the map label files, the final step to inserting maps into this program is setting up the correct buttons, if needed, to display different types of thematic maps of the same area. These modification are made on the "Preferences and Set Up" window, ID 146). Using SuperEdit, open the card in window ID 146. On it you will see a part of the card with the text "Type of Base Map", which has been set up to accommodate up to six associated buttons and their labels. The program needs to have all six button in it to run properly, so we DO NOT delete any of these, but rather drag the number we need to the visible part of the card. For example, if you have two different thematic map types to display, drag both "Map Type 1" and "Map Type 2" along with their associated labels into the area on the card set aside for them. Then, without deleting the labels associated with them (which have script attached) change the label to reflect the map content (in the Vector Tools, select the letter "A" and modify the text). for example , you might change these labels to "Topographic Map" and "Airphotos". These button are brought into view from buttons in the "Map Windows Control Palette" and in the "Map Features display" palette.

Adding Camera Icons to Map Overlays

Camera icons provide links (access) to images. Adding camera icons to Map images is almost identical to that of adding them to other Image Window Overlays, and is done while running the program. From SuperEdit, hit "cmd-R" to run the program in SuperCard, and then:

1) Open the image windows and map window to which you wish to add camera

icons. You can do this from the appropriate ClickList or, in the case of the map, from the Index Map window, after you've modified that to provide access to different map segments.

2) Use the File-Open Project Modifier menu to open the control window for making changes to the program (these include adding camera icons, moving overlay labels, and adding or modifying image window captions)

3) Click on the blue button entitled "Open 'Image/Photo Icon Maker' Window, and if needed or wanted, remove any icon from previous runs by answering "Yes" to the pop-up message.

4) After deciding where to put your icon, and which number window you are calling up. click on the green button at the center of the "compass" found in the Photo&Image Icon Maker Window, which is marked "Step 1".

5) Enter the number you want the icon to have. Be sure to enter three digits.

6) Next, select the directional arrow that best points in the direction the image is taken (or choose none, if you want). Answer "Yes" if you want the new icon's "Yellow Info" label to indicate that the new image will contain icons with access to other images.

7) Now, go to the "Finished Icons" window, and "mouse-over" the new icon, and note that the "yellow Info" label appears. This means you are in the browse mode. To be able to copy it you need to be in the "Pointer" mode, so click on the purple "Set Icon Window Tool to Pointer" button in the "Project Modifier" window.

8) Next select the icon(s) you want to copy by dragging around them or clicking on them, can copy using a "cmd-c".

9) Bring the window you want to paste icon(s) into to the front, and use "cmd-v" to paste. The icon will be placed in the same position (that is line and sample coordinates) it had in the Finished Icon window.

10) To move it to the desired location (and move any other overlay labels too) click on the "Arrow" in the Project Modifier window (near the top right side of that window).

11) Now click on and move icons around to position them, and click and use the delete key to delete unwanted icons, once they are selected.

12) Now, to save these labels back into the project's "Image Window Overlay" card for this image, click on the orange "Save Edited Overlay Back into Project" and answer "Yes" twice.

13) The window will save these new icons, along with all previous labels overlays back into the project, close the image window down, and reopen it for you to test the results.

STEPS TO ADDING IMAGES TO DDM.TEMPLATE

This assumes you have already created a label overlay for your image. The program will place this overlay on top of your image when it is displayed in DDM; the overlay can be turned visible and invisible using the show/hide button in the program. The label overlay might consist of lines, arrows, patterns and/or text etc.

Creating an Image Overlay

To create an overlay in Illustrator:

1) Start Illustrator
2) Open a "New" file
3) Use the "Place" command to insert your image on the background layer
4) Make a new layer in Illustrator in which to draw your overlay.
5) Place your labels/patterns, text, etc. , taking care to place a short vertical line, starting at 0,0 in the upper left hand corner of the image, to be used or alignment purposes by the program (that is to line up the overlay with the image when displayed by the program). Use the "pen" tool with the shift key down to draw straight lines (click twice, once in upper left, then below it about 10 pixels or 1/4 inch).
6) When done adding labels, we next need to save the labels layer in a format that SuperEdit can import. To do this, first delete your bitmapped image. Then, use the File-Export menu to save the file in a Pict format, and save the remaining file with the same name as your image file, except, instead of using the three letter extension "IMG", use "LBL" (this just to keep your self organized - you might also want to save these in a separate folder).
7) Quit Illustrator.

Adding the image's name to the Clicklist of Images

We do this so the program can call it up from disk. We assume you have already saved the image out of Photoshop into the folder "ImageProducts". If not please do this NOW. (Ideal image parameters are to save it as an 8-bit file, in a "Pict" format, with a size of about 800 x 600 pixels).

To add the Image's name:
1) Start SuperEdit
2) Open the project DDM.Template.99.xx.xx
3) Save the project with another name (I suggest with the current date or a lower case letter added to the date, if you've already saved a version of it using today's date).
4) Double-click on (to open) Wd ID 110 (Wd # 11), and then also double-click on the card (Cd ID 101 of Wd ID 110) to open it.
5) Click three times in the card's "Tool Type" selection box (upper left hand side) to select the "Field" Tools, then click on the letter "A", to select the "Text" tool within the field tools.
6) We need to make a line next with the data for your new image. either (a) find a line you wish to replace, scrolling down the field if needed, or (b) if there are none to replace, go to the next-to-the-last line on the list. The easy way to add a line is to copy the line above your line, and paste it back into the field, and then modify this line. Note that the last line MUST start with S#zz, or you won't be able to see the last line of "real" data so you have to insert new "last" lines above this S#zz line. Do either (a) or (b) above, and then make sure the line you want to modify is visible in your field.
7) In the Finder, go to the image file you wish to add to your project, click on it once, and copy the name. Note the last three letter of it's name MUST be "img".
8) Go to the line you're modifying, and select the name of the file you are replacing, and use "cmd-v" to insert the new name. If needed, be sure to

change the "S#" of the file to reflect adding a new image. Each image must have its own unique "S#". NOTE that the S# must correspond to the card number (NOT ID) below.

9) Add a key word identifying phrase that will help the user locate this file using the "find" or search capability in this window (which is more properly a palette). Remember this phrase is also attached to the window name when the image is displayed by the program.

10) Use "cmd-K' to both compact and save the program (do this "early & often" to preserve your work.

Note that you can edit the ClickList of Images while running the program in SuperCard (see after the next section).

Inserting the Image Overlay

Next, in the "Image Window Labels" file, we'll make a new card (or modify an old one) and insert the label overlay you've saved out of Illustrator into this card.

11) Next, double-click on (to open) Wd ID 115 (Wd # 16). We assume you will be modifying an already existing card here. If one does not exist, select the last one in the window, and use "cmd-c" to copy it. Then, click on the left side of the list of cards, just below the last card, to set insertion point for this new card, and use "cmd-v" to paste the copied card. * Note that each card has a graphic named "ImageNumber" near the upper left side. The number in this case corresponds to the card number contained in the window's list of cards (NOT the card ID!)

12) Now, check that the card you are modifying has a Card Number that matches the "S#" that you have assigned you new image in the "Clicklist of Images" field. (The Card Number is the left-most number in the list of cards in the Image Window Overlays, NOT Card ID, which is the number in the second column). Also, note S# must have three digits, thus card 1 corresponds to an "S#" of S#001).

13) Then double-click on the wanted card in the window "Image Window Labels" to open it. Use the "View-Show Invisibles" menu to show any invisible labels on this card.

14) Use "cmd-a" to select everything. Then use "cmd-u" to ungroup these objects. Now, we need to remove everything on this card except two objects in the upper-left. We do this by holding the shift key down, and dragging the mouse over both the "ImageNumber" label in the upper left side of the card, and the line which is in the extreme upper-left side of the card. This de-selects these two objects, leaving the others selected. Then we press the "Delete" key to remove the still selected objects.

15) To insert your labels, use the "File-Import Graphics" Menu. You'll get a note saying the graphics objects have been imported.

16) Use "cmd-v" to paste the label overlay into the card. If asked to add a "color-look-up table" say "No".

17) You may find letters that need changing, with font sizes that don't match, and colors that need to be modified. I often add a dark color rectangle to the card temporarily to aid in this work. After drawing one, use "cmd-=" to send it behind the labels in your overly, and don't forget to delete it when

you're done with it. If you want, you can also import the image to the card (and then subsequently deleted it) to see if your colors for the label are OK. If you do this, you might want to temporarily add the "CLUT" to the project and then delete it (note t be added later on how to do this).

18) To change colors of the text, select the "vector" tools option on the left side of the card, and the "Text" tool (letter "A", and select the text whose color you want to change. Go to the lower left of the window, and click on the left-most box and then on the left-pop-up drag the mouse to the color you want, and release.

19) Next, look at the upper left-hand card graphic "ImageNumber", and be sure it matches that of your card number. If not, select the "Vector Tools" associated with your open card (the upper left hand box, should be a pair of calipers), and blow it select the letter "A" to edit vector text objects. Click in the ImageNumber graphic and change the number to match the card number & "S#". Note that it must have three digits, thus for example, card 2 should correspond to S#002", and the ImageNumber label should be "002".

20) Lastly, after removing unwanted labels, and modifying everything to your happiness, you need to group your new additions to the card. To do this, use the key combinations "cmd-A" (select-all) followed by "cmd-G" (to group the selected objects) and then close the card.

21) Now use "cmd-k" to save/compact your project, and then "cmd-r" to go to SuperCard and run the program to check your work. Call-up your new image from the Control/Access - Open/Close List of Images menu, and double-click on the image name in the list.

Editing the ClickList of Images in SuperCard

1) While running your project in SuperCard, open the ClickList of Images, and from the File Menu, open the Project Modifier window.

2) In the lower right side of the Project Modifier window, click on the button "Enable Edit of ClickList of Images". Notice that two red warning signs are displayed, one in each window.

3) You can now edit the scrolling field in the ClickList of Images. while editing, you cannot single or double-click in the field to find images on the map or to call-up images.

4) When you're done editing the field, either click on the button "Save Changes to ClickList of Images" in the lower right of the Project Modifier window, or on the "Warning "sign in the ClickList of Images.

5) Normal function is now restored to the ClickList of Images.

Adding Camera Icons to Image Overlays

1) Open the image windows and map window to which you wish to add camera icons.

2) Use the File-Open Project Modifier menu to open the control window for making changes to the program (these include adding camera icons, moving overlay labels, and adding or modifying image window captions).

3) Click on the blue button entitled "Open 'Image/Photo Icon Maker' Window, and if needed or wanted, remove any icon from previous runs by answering "Yes" to the pop-up message.

4) After deciding where to put your icon, and which number window you are calling up. click on the green button at the center of the "compass" found in the Photo&Image Icon Maker Window, which is marked "Step 1".

5) Enter the number you want the icon to have. Be sure to enter three digits.

6) Next, select the directional arrow that best points in the direction the image is taken (or choose none, if you want). Answer "Yes" if you want the new icon's "Yellow Info" label to tell the user the new image will contain icons with access to other images.

7) Now, go to the "finished Icons" window, and "mouse-over" the new icon, and note that the "yellow Info" label appears. This means you are in the browse mode. To be able to copy it you need to be in the "Pointer" mode, so click on the purple "Set Icon Window Tool to Pointer" button in the "Project Modifier" window.

8) Next select the icon(s) you want to copy by dragging around them or clicking on them, can copy using a "cmd-c".

9) Bring the window you want to paste icon(s) into to the front, and use "cmd-v" to paste. The icon will be placed in the same position (that is line and sample coordinates) it had in the Finished Icon window.

10) To move it to the desired location (and move any other overlay labels too) click on the "Arrow" in the Project Modifier window (near the top right side of that window).

11) Now click on and move icons around to position them, and click and use the delete key to delete unwanted icons, once they are selected.

12) Now, to save these labels back into the project's "Image Window Overlay" card for this image, click on the orange "Save Edited Overlay Back into Project" and answer "Yes" twice.

13) The window will save these new icons, along with all previous labels overlays back into the project, close the image window down, and reopen it for you to test the results.

The DDM.Template includes a place for each figure to have a caption included; the procedure for adding these captions is at the end of the next major section.

ADDING TEXT & CAPTIONS TO THE TEMPLATE

Introduction:

There are two types of text to be added to the Template. The first is text data associated with any map, which may include a general text discussing the map, and/or one or more field trip(s) or guidebooks. The second is the text that is displayed in Image Window Captions. The procedure for adding caption text these is discussed below .

Adding "Guidebook" Text to the DDM.Template

General text, for example Guidebook articles, or that introducing a map is added in windows with IDs above 300. For example, text included with a given map can be added to a field on cards in window IDs 301 to 303. Additonal text, perhaps other fuidebook field trips can be added if wanted beyond those included in the Template (which includes wndows for four articles). The procedure for adding additional text windows these is beyond the scope of the present discussion. This program uses the convention of allowing text to be stored for three different levels of user, ranging from "For users without much science background" to those with a "Basic Geoscience" background. (Users can select the appropriate level from the Preferences Window, using the Apple Menu to access this). In order to set this "multi-level" user system up, any one given text article is stored in three successive windows. For example, a field guide to the Deerfield Basin Field Trip may be stored in fields on cards in windows ID 301, 302, and 303. The text that expresses this to those with a "Geoscience Background", that is, using language with the most advanced level of terminology would be stored in field on cards in window ID 301. The same field trip might be discussed in more general terminology in the fields stored on cards in window ID 302. Likewise, the most carefully worded and selected vocabulary, using the least sophisticated terminology would be stored in window ID 303. These will be automatically selected and displayed when the user changes "User Level". If a writer chooses to include only a single level of text, she should either (a) insert the same text into all three windows that apply to that text, or (b) disable the selection buttons for unused Levels in the preferences card.

To add text:

1) First change the name of the window to one describing your particular text (.g. "Guide to Tatara-San Pedro Volcano". To do this select (click once) on the window name listed in the "Project" window, and use cmd-i" keys and type the desired name into the pop-up window's name field. Do the same for the other two associated windows.
2) Note the window ID (<u>NOT</u> Window number) of your first or "Level 3" Window, which we will use next in adding this guide or text article to the "Clicklist of Field Trip Guides ".
3) Open the card in window ID 108 "Clicklist of Field Trip Guides", and select the Field Text Tool. either copy and alter a line or add a new one, bearing in mind that you must (a) not use any commas except those already in the lines, and (b) keep all columns justified.
4) Make the Guide# correspond to that of the Window ID (e.g. in the Template window ID 301 would be "G301", followed by two dashes (G301--); its name is "Introduction to Tatara San Pedro". Each entry must have a unique TripID, which must match that found in the ClickList of Maps (window ID 106). This is used to cross-reference hyper links between mouse-click in the text, and those on the map.
5) Enter any descriptive text needed to inform the user what they might find in this text article. Close this ClickList.

6) Open the Level 3 text window (e.g. here, 301 or the highest User Level. Open the first card. If there is already text in the card, use the Field Text Tool, click in the field, and use "cmd-a" to select all, followed by the delete key to remove the existing text,

7) Fire up you word processor, open your document, and select all text you wish to insert. Note that there is a 32 thousand (32K) character limit in any card field in SuperCard, so be sure to not try to paste more that into the field, but rather add the additional text to the next card (which is essentially page two). Copy you text and return to the card with the field into which you wish to add the text. Again, using the Field Text Tool, click in the field and use "cmd-v" to paste your text. If your text exceeds the 32 K limit, do the same for the next card, etc.

8) The convention in using this text field is that if you are viewing a map, and you single click on the text with the name of a photo icon (say S#003 for example), the program will center the map and blink the appropriate photo icon. Likewise for the text Stop 1. For those text links that refer to images, a double click will open the appropriate image (thus these lend themselves to text figures and tables too, if stored as images).

9) To facilitate the use of these, you as the author must make any Stop numbers and S# appear in bold text, and in addition you must make sure it is grouped. To do this select the text in the field using the Field Text Tool. then use "cmd-t" to obtain the pop-up "Text" window. click on the Style boxes Bold and Group, and then close the window. Do this for all needed hyper linked text.

10) More advanced uses can look at the script associated with these window (use "cmd-e" when the window (e.g. window id 301) is selected. Examining the script associated with the handlers "On mouseUp", and "On clickList" will suggest some additional options for linking, as explained in the text of that script for clickLink below:

```
on ClickLink
  global globThingToFind, GlobPhotoToGet
  --next script gets var click text from this field's script:
  --put the clicktext into globThingToFind
  --below we deal with case of text not being associated with a 'locate' or 'find' on the
map
  -- & thus is a special link that needs to be delt with for this wd only
  -- the default below assumes it deals with the map
  switch
    case char 1 to 2 of globThingToFind = "S#"
      put globThingToFind into GlobPhotoToGet -- takes care of reference to call up any
image
      send getCdName to cd fld id 551 of cd id 101 of wd id 101 -- Wd "Project Data Wd"
      exit clickLink
    case char 1 to 4 of globThingToFind = "Stop"
      --• I include this so that double click on Field Trip Stops aren't responded to
      exit clickLink
      --      •1999.12.03 -cdc- Below are three other examples:
      --      the first one opens a window with a series of cards
      --       to flip through show a process
      --      the second lets you hilite ertain text & call up an image window
      --      the third shows how to skip ahead to text further on in the text
      --      field on this on another card
      --    case globThingToFind = "C03"
      --      open wd id 155 --Berkshire Mountains Cross-Section Evolution
      --      exit clickLink
      --    case globThingToFind = "Fig. 11"
      --      put "S#195" into GlobPhotoToGet -- cd "GlaFig11.IMG"
      --      send getCdName to cd fld id 551 of cd id 101 of wd id 100
      --      exit clickLink
      --    case globThingToFind = "Skip Background - Go to Traverse"
      --      global globStringToFind
      --      put "RYANS  HILL TRAVERSE" into globStringToFind
      --      send FindThisString to bg btn "Find Text"
      --      exit clickLink
      --the below just opens the card shown
    case globThingToFind = "p.2"
      open cd 2
      exit clickLink
    case true
      answer "I couldn't figure out what to do with " & quote & globThingToFind
           & quote &" - Sorry!"
  end switch
end ClickLink
```

Editing Image Window Captions in SuperCard

1) While running the project in SuperCard (from SuperEdit, use the "cmd-r" key combination), use the File-Open Project Modifier menu to open the control window for making changes to the program (these include adding camera icons, moving overlay labels, and adding or modifying image window captions)

2) Inside the green box near the bottom of the window, click on the "Enable Image Caption Modification" button.

3) Open the Image whose caption you wish add or modify. Open the existing caption palette for this image window by option-clicking on the image. If no caption exists for this image, you will get only the name of the file, followed by two dashes, and a tab (which appears as a "space".

4) Click in the caption text field, either at the place you wish to edit, or, if adding new text <u>after</u> the tab (space). Actual editing capability is limited, but you can select and delete text, and type new text in.

5) When done adding or changing the text in the "Project Modifier" window, click on the red-rimmed orange button named "Save Edited Caption into this Project". If you are adding a new caption, you'll get a notice that no previous caption was in the project, and a new one is being added.

6) That's it! When you next open the caption window and this image is displayed, you'll see this new caption. Note that there are three different "User Levels", and that may entry using this method (described above in steps 1 to 5) will store the new caption into a field that corresponds to the user level being viewed. For more on this see below.

Adding Caption Text Using SuperEdit

7) For the more advanced user: The Image Caption Window (ID 117) contains two cards. The first is what opens when you option-click on an open Image window. The second contains fields where the data is stored, and from which any text is copied to be inserted into the first card for display on demand.

8) There are six caption text fields, best thought of as three groups of two. Thus the two top left-most fields contain text for Level 3 caption material, to their right, the next two contain Level 2 material, etc. The three levels give the user the option to display caption material at various levels of sophistication: level 3 is the most "advanced", and Level 1 is the least: I aim

Level 1 text at middle-school level readers. The level displayed is set in the Preferences Palette. If you wish to add or edit new text you can insert the new text or modified text directly into these text fields while in the SuperEdit environment. Simply copy the text from your Word document, and paste it into the corresponding field.

9) To add text directly into these fields in which there are existing captions, I recommend you first copy and paste all text out of both text fields of a given level into a Word document, and then run a sort on the document.

10) All data for a given image is store as a single paragraph (ending in a "return" character, thus the sort will arrange these caption by Image Name, making it easier to edit the captions.

11) Note that any caption field has a 32 K character limit, thus is you go beyond this limit, you must use both fields, each must contain fewer than 32 K characters. You can check the number of characters in any document in Word, by doing a "Word Count" found in the "Tools" Menu.

MAKING THUMBNAIL CARDS

Introduction

This secton discusses creating Thumbnail Window Cards, to be inserted into the Template's "Thumbnail Index of Images" Window. The goal of this process is to create a series of cards which contain 96 x 96 pixel size thumbnails of the Image (*.IMG) files included in your program. To do this, we use a program called "Quicknailer" to create the thumbnails, complete with file name beneath each, and save this as a pict file. Then we import that, along with your file names from you "ClickList of Images" window into the SuperCard program "DDM.ThumbnailFormater.2000.xx.xx". Run this program, which attaches the needed name and script to the objects (images and file names, along with "S#"), and then cut and paste the results from the card in the ThumbnailFormater into the appropriate cards in your DDM.Template program's "Thumbnail Index of Images" window. Lastly you take the Quicknailer Pict file, load it into Photoshop and cut it up into 8 x 5 groups of thumbnails and save each as a Pict file into the Thumbnail folder of your project.

Using Quicknailer to create the Thumbnails

1) Have your final selection of image files for this version of your DDM in your "ImageProducts" folder.

2) Start the Quicknailer program. On the pop-up window use the following settings (from QuickNailer v.1.7.1):

a) For type of file to produce, select Thumbnail

b) For the following "Show" boxes, select only "Thumbnail" and "File Names".

c) For Dimensions, select 96 (pixels).

d) For "Depth", select 16 (bit color)

e) Set "Background" to "White"

f) Set "Info" to "Black".

g) Set "Grid" to "None".

3) Drag your "ImageProducts" folder to the "Drag & Drop" folder icon.

4) Drag the window your thumbnails are on out so that it is eight images across.

5) Export the file as a "Contact Sheet - Macintosh Pict" file.

Formatting the Thumbnail Window Cards

1) Start SuperEdit, and load the program DDM.TNailFormater.2000.xx.xx. Save it with another name. Open the card in the second window (Thumbnail Overlay Maker Window), and use the Menu "File-Import Graphics" and select the thumbnails you created with Quicknailer.
2) Paste the graphics into the card, and while they are still all selected, slide all of the objects down so you have about 1/2 an inch between the top of the card and the upper-most thumbnail.
3) Now we have to remove the thumbnail images, which we will later format using Photoshop into card size pieces (8 x 5 thumbnails) and save outside the Template program to be loaded when running the program on demand (as pictureData). To remove the thumbnails, use the cmd-f key combination to open "Find" function dialog box. Under Object Types, select only "Paint Graphics". Under "Select Only" check the button "All". Hit the OK button, and when the objects are all selected, hit the delete button.
4) Next, go to your "Image ClickList" window (ID 110), and select all the text in the card field (to do this, select the Text Field Tool, click in the field once and use "cmd-a" to select all the text, and cmd-c" to copy it).
5) In the TNailer program, open the first card in the first (Instruction Window) window , and paste your text into the field on that card.
6) Now, start the program (use the "cmd-r" key combination), and click on the button "Open formatting window", and answer the dialog box by clicking on the "Newer DDMs".
7) Start the formatting by click on the button "Click to Format". Note that if your ClickList of Images doesn't jive with the images in your "ImageProducts" folder, you'll be notified, and you should figure out what's not in both.

Next, we'll cut and paste these scripted objects into their respective cards in your Template's "Thumbnail Images window". To do this quit the program, and open its second window in SuperEdit.

8) Select five rows of images by clicking on the right side of the card, just below the name of the fifth row, and dragging up to almost the "Example" graphic. Cut them by using the "cmd-x" keys. Notice that there are "invisible" objects that overlie where your "paint" or "bitmap" thumbnail images were, and be sure to include them in your selection.
9) Open you latest and greatest version of the Template, and open its window ID 151 (Thumbnail Index of Images). Open the first card and paste these object into it, being sure to center them, and to line up the bottom of the first image's file name with the line you see at about the Y=144 location (that is, 144 pixels down from the top).
10) Close this card, and repeat the process, putting the remaining rows on their respective cards. If you have extra cards, delete them from the Template. You can resize the background of the last card to conform to the number of rows you have if it is less than 5. To do this, with the card open, use "cmd-b" to toggle to the card's background. and with the View Menu, select "Show Card". Then click on the lower right corner of the background and drag it to shorten the card's length. You may need to move the gray background to do this good stuff (to see the card layer especially).

11) Notice, if you need additional cards, be sure to change the card number in the upper right, and to include the card's script. You must also modify it to reflect the third line in the "on openCard" handler to reflect opening the correct thumbnail image file (e.g. here if we are opening Thumbnail Index Window card 5, we need to change the script to open the image "ThumNail5.IMG as pictureData into the graphic "theImageBox".

Formatting the Thumbnail Images In Photoshop

Here we're creating a series of pict files, each with five rows of eight images in each row. These pict files will be cut out of the Quicknailer pict file., and save in the Thumbnail" folder.

1) Launch Photoshop, and under the Menu "File-Preferences-Units & Rulers", set your units to pixels. Then open the Pict file of thumbnails you created with Quicknailer.
2) Select the first five rows of images and cut them from this pict file.
3) Use "cmd-n" to open a new image, and set name to "ThumNail1.IMG" and it's size to 900 wide and 700 high. Paste your image into the new Photoshop window.
4) With the "move tool" (arrow icon) selected, drag a horizontal blue "marker" line down to 150 pixels. Do this by clicking in the ruler and dragging down.
5) Now click on your image and drag it to center it horizontally. Line up the bottom of the name of the upper row of files with the blue line at 150 pixels down from the top. This should assure the scripted graphics in the Thumbnail Index window line up with the image when it is called as pictureData into "theImageBox". If you have to adjust this you can do so with Photoshop and the project running at the same time, and adjust and then call up the image in the Template until you get it right.
6) To save this as a Pict file, use the "Save a copy..." option from the File Menu, and select "Pict" as your option. These files need to be in the Template's "thumbnail" folder to be accessed when the program is run.
7) Once you've saved the file, you can select all in this image and delete it, and use this Photoshop window to receive your next five rows of images. Be sure to change the name to reflect the correct card this new image will be displayed in. (E.g. The next Pict file that will be displayed in Thumbnail Index Window card 2 should be saved as "ThumNail2.IMG", etc.
8) That's it. Run the Template when you're sure you've got them done, and make any adjustments needed to the files in the "Thumbnail" folder.

==

Fine Tune the Project: Use of the Project Modifier Tools

The Project Modifier window has several components that will streamline the process of cross-indexing images with each other and the map, aid in placing new labels and sample sites on photos and the map, and allow you to edit Image Captions and the ClickList of Images.

The top one-half or so of the Project Modifier is devoted to adding and editing labels to Maps and Image overlays. These "labels" may be either feature names or samples sites that will be "click-able" so they can be used to display sample analytical data. They each can have a SuperCard name attribute added to them via a button provided in this window, so they be located by the program's search or "Locate" capability that compiles a ClickList of all images and maps on which a sample site or feature is found, to make them available to the user via this ClickList.

Features :
•**Hand or Browser Button** - click this to "run" the program, to try out your changes or to cancel the use of another tool (but only after you've saved changes back into the program - see next button description).

•**Save Edited Overlays Back into Project** button (Yellow & Blue): Use this to save the top window's overlay back into the project (this applies to Map or Image windows). All your additions will be added back into the appropriate card into the Overlay windows, and then the card will be re-opened so you can check the process. Failure to do will result in your changes being lost. Note that before you use this button, you should verify that the card field "Top Window's Name" above the button is identical to the one you wish to save; if it is not, click the green button (named "**Check Top Window Name & Tool**" just to the field's left of the field before saving the overlay back into the project.

•**"A" or "Text" Tool** button. This tool can be used to modify text labels already on overlays, or to add new text objects. The tool will continue to function until you click another tool. A word of caution: if you modify text objects that have already been "named", the name cannot be changed while running the program. Thus if you want the label to be "Located" by the program, it is best to make a new label if the one you are modifying is already named (see "Name New Graphics" below. (Note that if you use SuperEdit, you can of course modify this name).

•**Name New Graphics**. After you've created a new text graphic (see above), you can use this button on the Project Modifier window to give the new object a name that is identical to it's "textData" which is the attribute that SuperCard gives to the text you have written. If this is what you want, click on the button after you've added sample sites or feature names, and be sure to save it back into the project, or else the new names will not be kept in the project. Some details or background: When first created, an object has no name, and is known to the program only by it's ID (for example card graphic ID 110 of card ID 101 of window ID 120). I use this SuperCard name attribute to do two things: First, to look for associated data (if there is any in your project), so I can access and display it. In this case, if I want to see a specific chemical analysis I must be sure to keep the name unique among chemical samples, or I'll see the wrong data. thus since this button quite literally names the object exactly what the "TextData" is, you should take care to type the label for the sample site accurately, an include no additional "dashes" etc.!. Second, the program has the capability to "look" through all images and maps and find each sample site, and give you access to seeing it, but only if you attach the correct name to each object. Unnamed objects are invisible to the search routine; named ones can be opened and centered within an image or map window, and even blinked for ease in locating them.

•**"Enable Edits" Button**: After you've made a new text label (AKA text graphic), there are a number of buttons with which you can modify it. To use them, you select (click in the circle) of the "Enable Edits" button, then select (click once on) the graphic you wish to change. At this point the field above the "Enable Edits" button should fill with a graphic name on the top line (if it is named already) and the first part of a unique SuperCard ID on the second line. You can then click on the appropriate button to modify the graphic. Again, to save these into the project you need to use the yellow "Save Edited Overlays..." button, or it will be lost when you close the window.

Font Size & Background. These editing buttons include the following:
•**Trans** - or make the graphic "transparent"
•A "**Black**" or "**White**" **Letters** option
•**Unlock** button
•**G9** to **G36** to enable different sizes of Geneva Bold Fonts.
The first two are (I hope) straight-forward. The "Unlock" button should be used if a previously drawn graphic, which was locked, is resized (e.g. by adding letters or changing font size) and you need to reshape it or move it using the "Pointer" tool. The " Font Change" buttons gives you six different sizes of Geneva bold fonts to choose from.

•If you are adding sample sites to a Map or Image overlay, you may wish to use the "Click to add either sample site cross" button. This will give you the choice, via a pop-up window of adding either a black or white cross to the top window, which you can then move around using the pointer tool. These are draw graphic objects, not "text" or "field" "plus" signs, which, if a window is resized, can be relocated in incorrect positions (especially if the text is right or left justified).

Other Access & Modification Switches.

•"**Open Image/Photo Icon Maker Window**" button. For use of this, please see the details in the above section entitled "To Add Camera Icons to Map and/or Image Overlays:" These instructions include the use of the purple and orange buttons below this blue button. Below are some general hints, useful to aid in placing image photo icons on the map. I recommend you open the "Thumbnail Index of Images" Window, and refer to it to see what images you need to plot on the map. If you grab the lower right corner & drag it to a smaller size, you can scroll around that window to keep the thumbnail Image in sight and still see the map, to help you in placing photo Icons. When you use the blue "Previous" and "Next" arrows in the Thumbnail Index of Images window to turn to the next image, the window will automatically resize to full size.

•**Enable Image Caption Modification**. To use this button, and its associated orange color coded button "Save Edited Image Caption..." see steps 1 to 5 of the above section entitled **Editing Image Window Captions:** The intent of these is to allow you, while running the program to modify and edit captions (or add new one if need be).

• "**Enable Edit of ClickList of Images**" Use this, and it's associated Save button with caution, because the program relies on the file names in this list to open the correct file and its associated overlay. A click on the "Enable" button changes the type of clicklist this is, and gives you the capability to edit within it. This is very useful on the fly, because the image's window is named by the phrase associated with the image number (that on the right hand size of the clicklist. I recommend you choose the first word carefully, with the thought that the ClickList can be resorted by the first word in this last column, thus an appropriate word that might be looked for, is most useful. For example, if the image has a green amphibolite in it, and you want to call attention to the fact that it is an amphibolite, a the 'green-ness" is not all that important, you might want to start the last column with the word "amphibolite" and not "Green". Note that you will not be able to use the "ClickList" in a standard way until you "Save the Changes..." which changes the type of ClickList it is, and locks the text from further change.
===

Using Cards as Animation or Sequence Players
I have set up two windows which show how one may use them to make animated "cartoons". These windows (ID 158 and 159) have the buttons installed to allow one to play back all cards in the window in sequence, set the time each is paused for viewing, and to single click from card to card to slow action down. These could be easily duplicated and new ones created and placed in window IDs after window ID 164 but before window ID 200 (which is where from window 200 to 300, I display analytical data if the project includes such.

Using PictureData Images as Animations or Sequence Players
I have inserted two windows named "Animated Using PictureData Window 1" and 2, which contain the code and elements so a Template user can modify them slightly to play sequential images through "pictureData" directly from a directory on the hard disk or CD. The file names for the images are to be stored in a background field of the card, and the directory or folder they are resident in should be included at the same level as the Map and Image Products folders. I recommend clicking through the script to work with it in the "trace" mode.

ADDING DATA ANALYTICAL TO A MAP (AND MODIFYING DDM.TEMPLATE FOR USE WITHOUT DATA)

Modifications of Template for Use Without Data

The DDM.Template has been designed to be used with data included, but with a few simple steps can be modified so that the same basic project can be used without data included. A critical line in the Project Script needs to be changed if you are NOT INCLUDING analytical data. Look for the following lines:

```
  --•••SET below GlobHasData TO FALSE IF NOT INCLUDING
ANALYTICAL DATA
  put true into GlobHasData
```

In line five, (shown just above here in this text) of the "on OpenProject" handler, you must put "false" into GlobHasData. What this does is bypass any data acquisition subroutines called for by mouse-clicks in the map or image windows (except for those to scripted objects, like photo icons) or for field trip text).

Menus added to display analytical data include the following two items, which should be deleted in projects in which data is not included. They are the last two menu items in the Menu "Controls/Access" entitled:

•Open/Close Sample ClickList Access (ID 107, item ID 110)
•Open/Close Data Display Switch Menu (ID 107, item ID 113)
•Save Data To Disk... (Item ID 103 of Menu "File" -I D 102)

Windows added to display analytical data include:

•Data Display Switch - window ID 105
•Save Data To Disk - window ID 113
•Data display windows with ID 200 to ID 300

These can be removed if you wish, or kept in; as long as you don't call for them, they shouldn't get in your way, if you remove Menu access to them. Note that although we use data fields to store analytical data in window ID 100 (the first window of this project), it is also used for several other functions in the program, so it cannot be removed even if you are not including analytical data with your DDM.

THIS ENDS THE MODIFICATIONS NEEDED TO USE THE DDM.TEMPLATE WITHOUT INCLUDING ANALYTICAL DATA.
==

How Analytical Data Is Stored, Accessed and Displayed

Analytical data can be displayed by this program, because any graphic item's name can be picked up by the program with a mouse-click to either the Map Window (ID 120) or one of the Image windows (IDs 131 to 139). This name (stored as the global variable "globDataWanted") can then be sent to script for further use. I store both the access and display script, and any analytical data in the first card in the first window of the program, known as the "Analytical Data" Window. An almost identical process goes on with all types of data, which are stored in card fields with unique names, with up to 32K characters possible in each. The key is to uniquely identify the data within a field (see section below on details) , and copy it to a variable to be inserted into a display field with in a card and window designed for this use. I do this in most cases by placing data in lines within a field, with the sample or site name first on that line, and separated by either a comma, space or tab from the data associated with it. Then I search the field for a line containing the sample, and after finding it, place that line (or series of lines if the data comprises more than one line) into the display card field. Because many studies have several types of analytical data, which might be taken from sites with identical IDs, I funnel all data through a "filter" or handler (subroutine) which checks to see what type of data the viewer wants to display. The script (the "getData" handler of card graphic "AllDataAccessScripts" of card id 101 of window id 100) looks for which buttons are "hilited" within window ID 105 (the "Data Display Switch"window) to decide which fields from which to search and display data. The handler then looks only in those fields to display it (this script is found in the AllDataAccessScript" graphic). It then calls the "handlers" found in the unique script designed to display the type of data contained in each field. See the above handler to trace the script to the appropriate graphic items which are labels with the type of data or field they get data from. Once copied to a variable, the data is then placed in appropriate display palettes, which range from ID 200 to 300; the particular palettes used can be found in each handler, and are titled in their window name appropriately. Note that these palettes are checked and made invisible when a sample site is searched for using the "Find" (command-F) and then made visible again, thus adding palettes outside this range of window IDs would frustrate this process. The first of each data type's display palettes also contains a button that is linked to open a card of the "Auxiliary Information - 2" (window ID 155) to qualify the data: how it was analyzed and collected. Note it is at the start of the handler "getData" that DDMs without analytical data exit data retrieval, because the global variable GlobHasData has been set to false in those projects (in the Project Script). If it is true, the process of data retrieval continues.

The "Sample Access ClickLists"

In addition, I also include for each type of analytical data, a "ClickList" which includes the sample site number, and if appropriate the associated geologic unit. This list I compile directly from the data fields, using a word processor to cut the sample number (and units) out from the list, and inserting them into the palettes. These palettes are called up from buttons in window ID 200, the "Sample Access ClickLists ". These lists allow the user to either locate the data sample site on the map (with a single click to the sample ID in the clicklist), or to display the data (with a double-click on the list). Failure to find the sample on the open map will give the user the option to have the project search all unopened map segments or images to find those which contain that site, and provide an access clicklist of those locations for the user display those maps or images.

Data Storage Fields and Associated Display Palettes

Included in the Template project is three types of unique fields for use in storing data, and four different ways these data are displayed. Types of display palettes include six rectangular palettes for displaying unit descriptions, eight major element geochemical palettes, with columnar display, a single table that scrolls to a given sample (The Age Data" Palette), and three groups of 9 palettes each that display in a horizontal sequence of adjacent palettes. Below I describe the display palettes and how they tie into the storage fields in the project's first window named "Analytical Data".

•**UnitDes1** (to UnitDes6) fields: These are designed to contain geological unit descriptions, with each description divided into seven lines (with or paragraph ends). The unit map symbol is found in the first line and is followed directly by two dashes; in no other place in this data set should that be true except in the first line associated with this description. The program searches for and finds the line number with this unique set of symbols, and then copies it, and the next six lines, to a variable and then displays it in fields in palettes 201 to 206, opening the lowest ID number of these that is not open at the time.

•**ChemDataFld1** (two fields, the second is named ChemDataFld2). These two fields contain data in single lines, in comma delimited format. Once found, the data is read into a vertical card field so that the petrologist can examine the data in a series of such fields in windows ID 211 to 219. I hand the global variable with the site's "short name" to the handler "GetChem" and change it to the following: [return & globDataWanted & ","]. By doing that I can uniquely identify the site ID as a complete site, thus for example, if I want to display site EMU.1's data, it will not by error display site EMU.1.1's data, because the data searched for requires a line return to start it, and a comma to end it (again, see the below section Uniquely Identifying Data). The data is parsed into lines with spaces padding it to align the data in a variable and then displayed in each palette's field.

•**Horizontal Data Display Palettes.** These include groups of 8 (or 9) palettes that are linked to the similar analytical data storage fields. Display palettes with window IDs 220 to 260 each start with an "header" or wider palette that has to have the header data inserted while in SuperEdit to coincide with the columns of data stored in their respective data storage field in window ID 100. As in the other data fields, the line of data is retrieved and displayed sequentially until all palettes are in use, at which point the user is then told to close one of that type of data's palettes to display further data. I have included complete script for the data storage fields PMagMasterData (one data field) and the two-field group AnalyADataFld1 and AnalyADataFld2 (two fields are used to store more than 32K of data). The script to call data from these fields are found in the card graphics "PMagDataAccessScripts" (handler "getPMag") and graphic "AnalADataAccessScripts" (handler "getDataA") respectively . The PMag palettes have Window IDs from 221 to 229, with it's Sample Access ClickList, named "GetPMag", with an ID of 230. Analysis A has palettes with IDs from

231 to 238, with an Sample (Data) Access ClickList (GetAnalysisA) with ID of 239. Although not tested I have created storage fields in window ID 100, script graphics and display palettes for analyses B, C and D data sets. Expansion of this can be done by duplicating the AnalDDataAccessScript graphic, altering it to reflect the new data set, adding two storage fields of the correct name, and changing the window ID's to display to those for AnalysisE, and renaming the existing display palette windows in the Template, using the "D" version as a guide to how to do that..

•**The Scrolling Display Palette** (or here "Age Data" Palette). This is a unique in that the analytical data is not store in the first window of the project, but rather within this window itself. The "getAge" handler is called in the card graphic "AgeDataAccessScripts" of the first window, and opens the scrolling "Age Data" window and sets the scroll to the line with the data in the field that has that sample's site number in it.

Uniquely Identifying Data:

Note: To provide unique access to data, and to save it to a file outside this application, we save a header or first line of data with the column headings, with each entry either a tab or comma in front of the data title. Likewise the data are saved with the "tab" or comma just in front of the data, as is the case for any data set saved out of a spreadsheet application. What is essential is that the header precede the data, and that a single identifiable character (a space or " ", a tab or a comma) follow the sample id, because that is what the program searches for in finding a unique sample to be sure it gets the entire sample's name and only this sample's name. For example, if we were putting together a table with trace element data into it, it might look like this:

```
Sample              Unit    Nb  Zr   Sr   Zn
BP.1.1                 -   9.3 261  363   64
BP.2               -  8.9 260  364   67
BP.1                   -   7.3 361  263   44
```

To uniquely find the data BP.1, we tell the program to find the string [return followed by the sample id [bp.1} followed by space or " ". Otherwise the program would first see the line with the data BP.1.1 and find BP.1 in it and return that line.

Using the Data Formatter Program

SuperCard doesn't have any way to align columns (no tab alignment function), so to display data in aligned columns, I use non-proportional fonts (Courier or Monaco) and put spaces in the correct location to align columns.   I have written a program (named Pad&AlignColumns2000.02.25d) to handle most of that work for you. Use it with caution, however, because it won't handle all the possible situations one can imagine. Input for the program is a tab delimited data set.   To use the Pad&AlignColumns2000.02.25d program, open it in SuperEdit and cut your data from a text application, and paste it into the upper field on the only card in the only window of the program. Then a click on one of the three buttons should give you aligned data in the lower card field, if all goes well. At that point, you can return to SuperEdit, and cut the data out of your field and paste it into you DDM.