

**Software Engineering Plan for**

**MONTAGE**

*An Astronomical Image Mosaic Service for the National  
Virtual Observatory*

**Version 1.0 (May 31, 2002)**

*This Plan is based on IEEE 1058.1-1987*



MONTAGE SOFTWARE ENGINEERING PLAN REVISION HISTORY

<b>Description of Revision</b>	<b>Date</b>
Initial Release – Version 1.0	May 31, 2002

## Table of Contents

1	Purpose of Montage .....	5
2	Project Deliverables .....	5
3	Project Milestones.....	6
4	Heritage Software and Evolution of the Astronomical Image Mosaics .....	10
5	Science Drivers and Use Cases for Montage.....	11
5.1	The Value of Montage in Astronomy .....	11
5.2	Technical Priorities for Montage .....	11
5.3	“Top Ten” Science Drivers.....	12
5.4	Example Use Cases.....	13
6	Design Elements of Montage.....	14
	Figure 1: Design Components of Montage.....	14
7	Major Areas of Effort.....	17
7.1	Interface with Astronomical Community: Science Goals .....	17
7.2	Interface with 2MASS, DPOSS and SDSS Projects.....	17
7.3	Deployment within the Architecture of the National Virtual Observatory.....	17
7.4	Development of Montage .....	17
7.5	Visualization of Image Mosaics.....	18
8	Project Structure.....	18
8.1	Project Life Cycle Model.....	18
8.2	Project Responsibilities.....	18
8.3	Project Interfaces .....	19
8.4	Management Approach.....	21
9	Technical Approach .....	23
9.1	Assumptions and Constraints.....	23
9.2	Use of Open Source Software.....	23
9.3	Portability of Montage Software.....	23
9.4	Delivery of Maintainable Code.....	24
9.5	Activities, Tools and Products .....	24
9.6	Build Strategy .....	24
9.7	Maintenance Strategy.....	25
10	Risk Mitigation Strategy.....	25
10.1	Definition of Risk .....	25
10.2	Organizational Responsibilities .....	25
10.3	Schedule for Baselineing and Updating Project Risk.....	25
10.4	Methodology for Risk Identification & Mitigation Plans.....	26
10.5	Risk Tracking.....	26
11	Configuration Management .....	26
11.1	Items Subject to Configuration Control.....	26
11.2	Organizational Responsibilities .....	27
11.3	Configuration Management Tools.....	27
11.4	Configuration Management Procedures for Documentation .....	28
11.5	Configuration Management Procedures for the Public Web Site.....	28
11.6	Configuration Management Procedures for Database Schema .....	29
11.7	Configuration Management Procedures for Test Datasets .....	29

11.8	Configuration Management Procedures for Software .....	30
12	Quality Assurance.....	32
12.1	Organizational Responsibilities .....	33
12.2	Quality Assurance Activities .....	33
12.3	Quality Assurance Tools.....	35
13	References.....	39
14	Acronyms.....	39

# 1 Purpose of Montage

Montage is an exemplar compute-intensive service for the National Virtual Observatory (NVO). It will deliver on demand *science-grade* astronomical image mosaics that satisfy user-specified parameters of projection, coordinates, size, rotation and spatial sampling. *Science-grade* in this context means that the impact of the removal of terrestrial and instrumental backgrounds on the fidelity of the data is understood over different spatial scales. The service will deliver mosaics of images released by the 2 Micron All Sky Survey (2MASS), the Digital Palomar Sky Survey (DPOSS) and Sloan Digital Sky Survey (SDSS) projects.

The computing challenge of Montage is to sustain a throughput of 30 square degrees (e.g. thirty 1 degree x 1 degree mosaics, one 5.4 degrees x 5.4 degrees mosaic, etc.) per minute on a 1024 x 400 MHz R12K Processor Origin 3000 *or machine equivalent*. Montage represents an evolution of a baseline engine deployed at JPL, *yourSky*. Incremental deliveries in 2003 and 2004 will progressively improve the science quality, speed and portability of the baseline code. The final deployment will be in January 2005.

Montage will run operationally on the *TeraGrid*, (<http://www.npaci.edu/teragrid/>). Users will submit requests to Montage through existing astronomical portals, and visualize and interact with the delivered mosaics through the same portals.

A fully documented, portable version of the software will be publicly available for running on local clusters and individual workstations. The compute engine will be quite general and will allow users to build mosaics from their own FITS format data.

# 2 Project Deliverables

**Table 1: Montage Project Deliverables and Release Dates**

Description	Release Date
Software Engineering Plan, including <ul style="list-style-type: none"><li>• Science Use cases and detailed science requirements, approved by the Customer Review Board (CRB). The CRB will be specifically asked to provide guidance on the spatial scales and pixel sampling frequencies over which science grade mosaics should be served.</li><li>• Software Engineering Plan, including<ul style="list-style-type: none"><li>• Software Requirements</li><li>• Technical Approach</li><li>• Management Approach</li></ul></li></ul>	May 31, 2002

<ul style="list-style-type: none"> <li>• Quality Assurance</li> <li>• Configuration Management</li> <li>• Risk Management Strategy</li> </ul>	
Public web page	May 31, 2002, with updates thereafter
Baseline performance of image mosaic engine & background removal server. Deliver baseline mosaic & background removal code.	June 30, 2002
Statement of Design Policy for Interoperability and Community Delivery	July 31, 2002
Annual Reports delivered to web page	August 31, 2002 August 30, 2003
MONTAGE: an image mosaic service deployed on the <i>TeraGrid</i> that will generate and deliver science-grade, image mosaics of 2MASS, DPOSS and SDSS.	<i>Staged deliveries:</i> February 28, 2003 August 30, 2003 August 15, 2004 January 10, 2005
Portable version of the MONTAGE software, available for download from the project web site. This includes source code as well as binary versions of the Montage modules for selected common platforms (including Solaris, Linux, and AIX).	<i>Staged deliveries:</i> February 28, 2003 August 30, 2003 August 15, 2004 January 10, 2005
A Users Guide that will include: <ul style="list-style-type: none"> <li>• Technical description of the service.</li> <li>• Instructions for building and testing the system on a user's machine</li> <li>• Maintenance Plan</li> <li>• Description of the background removal algorithms, and their impact on the scientific integrity of the calibrated images.</li> <li>• Design</li> </ul>	<i>Staged deliveries:</i> February 28, 2003 August 30, 2003 August 15, 2004 January 10, 2005
Final Project Report delivered to web site	February 15, 2005

### 3 Project Milestones

#### A) Software engineering plan completed. Due Date: May 31, 2002

- Publish the Project Management plan on a project web page. The plan will include the following:
  - Science use cases and detailed requirements, approved by the Customer Review Board (CRB). The CRB will be specifically asked to provide guidance on the

spatial scales and pixel sampling frequencies over which science grade mosaics should be served.

- Technical Approach
  - Software Engineering plan
  - Management Approach
  - Quality Assurance
  - Configuration Management
- Deliver a Risk Management strategy that will be available internally to the project and the sponsor.

### **E) Code baseline delivered. Due Date: June 30, 2002**

Describe the physical problem that is addressed by the baseline code. The baseline code will include the ability to remove backgrounds through application of a low pass filter.

#### ***YourSky* Mosaic Service**

- Baseline the performance of the *yourSky* image mosaic service, which offers custom access to mosaics of user-selected size, coordinate system, and projection, with bilinear sampling. The performance will be baselined on the JPL PowerOnyx, equipped with 8 x 194 MHz MIPS R10K processors and 2GB RAM. Performance and scaling curves will be published on the project web page. Documented source code will be delivered to the project web page.

#### **Background Removal Service**

- Baseline a general background removal service that provides background removal decoupled from the mosaic service. Perform a global fit to the 2MASS and DPOSS data with a high-pass filter that adjusts images to a common sky model but preserves background flux. Write background parameters to a meta-database. Publish baseline performance to fit the images in those parts of the sky publicly released by 2MASS and DPOSS. Documented source code will be delivered to the project web page.

All software referred to above will be made available to NASA ESTO-CT contact scientists.

Publish, on the project web page, initial system requirements and design documents.

### **H) Design Policy for Interoperability and Community Delivery. Due Date: July 31, 2002**

Declare that this project will use Virtual Sky and OASIS as its primary clients. The API for other clients to run the mosaic engine will be published on the project web page. Declare that the operational version of the mosaic engine, *yourSky*, will run on the NPACI *TeraGrid*, and that the project will serve mosaics from three surveys, 2MASS, DPOSS and SDSS. Under the guidance of the Customer Release Board, we will establish

policies for the public release of the service and for naming customers who will be permitted access to the service on a shared-risk basis before public release. Declare working relationships with collaborators at NPACI (SDSC), 2MASS, DPOSS and SDSS.

The Design Policy for Interoperability and Community Delivery will be published on the project web site.

Publish on the Project Web page updated requirements and design documents and initial test plans.

**B) First Annual Report delivered to project web site. Due Date: August 31, 2002**

Deliver report to the project web page.

**F) First Code Improvement demonstrated. Due Date: February 28, 2003**

Develop Science Grade Mosaics that conserve energy and support background removal, with metrics specified through the guidance of the CRB and scientifically validated under its auspices. Access to this service will be through a modification of the existing *yourSky* web form.

***Mosaic Engine*** –

- Ensure conservation of energy in mosaics
- Handle image rotations in all WCS projections
- Metric: The following metrics apply to science grade mosaics; their precise values will be established through the guidance of the CRB:
  - Reduction in the average deviation from the measured energy per unit area (we anticipate roughly 50%) when constructing mosaics in at least 10 WCS projections with any image rotation.
  - Spatial scale of mosaics and spatial re-sampling of pixels that allow science analysis (we anticipate 1 to 5 degrees spatial scale; and full, 1/2, 1/4, and 1/8 resolutions).
- Apply Background Removal Parameters that support background subtraction models:
  - ***Common sky model that preserves total flux***
  - Preserve point sources only
  - Preserves feature on a scale that allows science analysis (we anticipate 1 to 5 degrees, as noted above).

Documented source code made publicly available via the project web site.

**I) Interoperability Prototype. Due Date: July 30, 2003**



- The improved *yourSky* code delivered in Milestone F) will run on the *TeraGrid* Linux cluster. Performance comparison between the PowerOnyx and the *TeraGrid* will be published on the web page.
- The improved *yourSky* code delivered in Milestone F) and running on the *TeraGrid* will be interoperable with the *OASIS* and *VirtualSky* clients, in that users place an order for a custom mosaic through these clients, receive notification of the completion of the request, and are able visualize the images.

### **C) Second Annual Report. Due Date: August 30, 2003**

Deliver second annual report to project web site.

### **G) Second Code Improvement. Due Date: February 28, 2004**

#### *Code Improvements*

- The improved *yourSky* code per milestone I) will run on the *TeraGrid*. The achievable computational speed-up will depend on the performance of the *TeraGrid* as deployed. We propose two performance metrics: A target computation speedup that ignores I/O time and a target overall speedup that includes both computation and I/O times. We will achieve a target performance that is *equivalent* to a *computation* speedup of 64 and an *overall* speedup, including I/O, of 32, for a 5 degree x 5 degree 2MASS mosaic (which will be the most computation intensive dataset) on a 128x1GHz (128 GFlops) target machine with a *sustained* bandwidth to disk of 160 MB/sec.
- Cache results locally for commonly requested regions. Develop cache of at least 2 TB. Metric: Demonstrate speed-up when cached mosaic is requested. Publish speed-up figures.

Documented source code will be made publicly available via the project web site.

### **J) Full interoperability. Due Date: August 15, 2004**

- Demonstrate that the compute engine accepts requests from the *OASIS* and *VirtualSky* clients for mosaics from the 2MASS, DPOSS and SDSS surveys, processes the request (includes accessing cached images as necessary), notifies the user regarding the status and availability of a mosaic, which can be visualized by the user. Visualization includes full user control of the image in real - time: pan/zoom, cropping, scaling, resampling, color table, stretch, and histogram equalization.
- Publish on the project web site updated requirements & design docs, and updated test plan and test reports, and a draft Users' Guide.

### **K) Customer Delivery. Due Date: Jan 10, 2005**

- Community delivery of the fully operational service. Deployment will be announced at a major national astronomical conference, where we will:
- Demonstrate custom mosaics from 2MASS, DPOSS and SDSS with sustained throughput of 30 square degrees (e.g. thirty 1 degree x 1 degree mosaics, one 5.4 degrees x 5.4 degrees mosaic, etc.) per minute on a 1024x400Mhz R12K Processor Origin 3000 *or machine equivalent* with sustained bandwidth to disk of 160 MB/sec and all input data previously staged on these disks.
- Present scaling curves specifying the variation in compute time with mosaic size and number of user requests
- Deliver a Users' Guide that will include descriptions of how to access the service, known caveats and limitations, and a thorough description of the impact of background removal on the scientific integrity of the images.

Documented source code will be made publicly available via the project web site.

Publish on the project web page final versions of requirements, design, test plan and test results. Deliver to the project web page an updated Users' Guide and a Maintenance Guide.

### **D) Final Report. Due Date: February 15, 2005**

Delivery of final report to web site.

## **4 Heritage Software and Evolution of the Astronomical Image Mosaics**

Montage represents an evolution of two image mosaic services that are currently available to astronomers. The JPL *yourSky* (<http://yourSky.jpl.nasa.gov>) portal builds custom mosaics of 2MASS and DPOSS according to user-specified parameters of coordinates, projection, sampling, and size. It employs parallel processing on an 8 processor SGI PowerOnyx, and accesses the 2MASS Atlas images from the Storage Resource Broker (SRB) at San Diego Supercomputer Center and the DPOSS images from the High Performance Storage System (HPSS) at Caltech's Center for Advanced Computing Research (CACR). An email notifies the user when the mosaic is ready for download. While *yourSky* delivers mosaics on demand, *VirtualSky* (<http://virtualsky.org>) offers visualization of pre-computed mosaics of DPOSS and several other surveys (NVSS, ROSAT. . .), in some cases covering the full sky. Neither service delivers science-grade mosaics nor portable code nor data fusion/analysis capabilities that are requisite to astronomical research. The OASIS applet (<http://irsa.ipac.caltech.edu/applications/OASIS>) does offer these capabilities, but without access to a mosaic engine.

Montage will address these limitations: it will deliver science-grade astronomical mosaics, will be callable from existing astronomical portals –including *yourSky*, *VirtualSky* and OASIS, and will be portable and available for download.

## **5 Science Drivers and Use Cases for Montage**

The technical priorities, science drivers and use cases described here are the basis for the Montage software development plan. This section has been prepared with the assistance of the Customer Review Board (section 7.3)

### **5.1 The Value of Montage in Astronomy**

The greatest return from Montage will be in enabling multi-wavelength studies of image mosaics. Analyzing and identifying astrophysical sources over a broad-wavelength range are one of the most powerful tools in the astronomer’s arsenal, requisite for scientific research and for observation planning. The analysis of pan-chromatic images has, however, been frustrated for two reasons: many sources are extended on scales of 1 degree or more, larger than the size of most images, and images at different wavelengths have different projections and spatial sampling. Montage will remove these impediments by delivering science-grade mosaics from multiple sources on a common, user-selected projections and sampling. This capability will support one of the most important scientific aims of the NVO, that of studying the pan-chromatic universe.

### **5.2 Technical Priorities for Montage**

- Deliver mosaics which have accurate astrometry, which preserve the photometric accuracy the input images, and which are free of any features introduced by the mosaic engine itself.
  - Involve the community in validation of these mosaics during development of the first release.
  - Many of the individual components of Montage will themselves be valuable tools in astronomy. For instance, astronomy does not yet have a standard image reprojection library. Where appropriate, the components should be offered to the community as deliverable products.
- Deploy physically realistic background removal algorithms to preserve the science quality of the image. Observation planning will often need only a flat background rather than true calibration quality.
- Ensure existing portals permit visualization and manipulation of the image mosaics
- Deploy truly on-demand mosaics through a fast service, with complete user control over the size, projection, coordinates, rotation and spatial sampling of the returned mosaic.

### 5.3 “Top Ten” Science Drivers

The list below is a “top-ten” list of investigations that can be performed with image mosaics. “Investigations” refers to:

- Morphology (size, shape & symmetry),
- Aperture photometry and fluxes, including upper limits. The latter will be one of the most important applications of mosaics, for this information can only be obtained from images.
- Position (astrometry) and geometry, profiles, color, and variability therein.

Many of the studies proposed here are explicitly called out in the report of the NASA NVO Science Definition Team [1].

1. study large and coherent extended objects (e.g., galaxies, reflection nebulae, Planetary nebulae, SNR)
2. study objects that are otherwise difficult to catalog, due to their surface brightness or complexity (e.g., diffuse or nearly invisible nebulosity, globular clusters, star formation regions like Orion)
3. study galaxy clusters or mixed-morphology large-angular scale objects (e.g., Virgo cluster)
4. study the larger environs of target objects (e.g., the cluster environment of galaxies; the GMC environment of a YSO)
5. study large scale structure (LSS) of the Universe
6. study very large amorphous regions of the sky (e.g., Galactic Center)
7. study the astrometric properties of the sky, such as motions of stars and ephemeral Solar System objects; astronomical lensing, and even camera/optics distortions.
8. study objects using multi-wavelength imaging; build 2-dimensional (spatial) SED planes (e.g., image/spectral cubes).
9. build ever larger mosaics, ultimately reaching the "all sky" scale –all sky images enable a unique form of science -- study of the Milky Way as a whole (consider the 2MASS all sky, where you can see the relationship of the local group, LMC/SMC, with respect to the Milky Way); finder "charts" or maps of the sky fall under this category.

10. Serendipity -- to discover new large-scale objects that were previously unknown and/or impossible to detect with small images (e.g., dwarf local-group galaxies, nearby low surface brightness galaxies, supernova remnants, LSS)

## 5.4 Example Use Cases

### Use Case I - Science Analysis

The SIRTf First Look ancillary VLA image is a 2x2 degree radio image of a field that will be observed by SIRTf. As a field uncluttered by galactic radiation in SIRTf's continuous viewing zone, it is a prime candidate for deep imaging of extragalactic sources. The VLA image contains many radio "blobs," many of which appear to be interesting and perhaps bizarre objects. Interpretation of these objects requires multi-wavelength measurements on a common projection and spatial scale. DPOSS, SDSS and 2MASS provide the broad wavelength base for analysis of these objects, yet analysis is tedious and error prone because the images delivered by these projects have different spatial resolutions, coordinates and projections. MONTAGE will eliminate these difficulties by delivering mosaics from these data sets at a common resolution, projection and in a common coordinate system.

### Use Case II – Observation Planning

The Multiband Imaging Photometer (MIPS) aboard the Space Infra Red Telescope Facility ([http://sirtf.caltech.edu/SSC/MIPS/mips\\_intro.html](http://sirtf.caltech.edu/SSC/MIPS/mips_intro.html)) has a scan length of 0.5°. Observations with MIPS must avoid bright sources that will saturate the detector, and is normally done by identifying infrared sources on 2MASS images. This is at present difficult to do because the 2MASS images are 512 x 1024 arcsec on a side and the effects of background variation from image to image complicate identification of sources in a consistent way. . Mosaics of 2MASS images that have a flat background (not necessarily science grade) will make the task of identifying bright sources much easier to perform.

### Use Case III – Science Product Generation

The Galactic Legacy Infrared Midplane Survey Extraordinaire (GLIMPSE) will use the Space Infra Red Telescope Facility (SIRTf) Infra Red Array Camera (IRAC) ([http://sirtf.caltech.edu/SSC/IRAC/SSC\\_B4.html](http://sirtf.caltech.edu/SSC/IRAC/SSC_B4.html)) to survey approximately 220 square degrees of the Galactic plane, covering a latitude range of  $\pm 1^\circ$ , and a longitude range of  $\text{abs}(l)=10-65^\circ$ . GLIMPSE will be a confusion-limited (approximately 300  $\mu\text{Jy}$ ) in the four IRAC bands. The survey will produce several hundred GB of data in the form of catalogs and images, which will be delivered to the SIRTf Science Center for dissemination to the entire astronomical community. The GLIMPSE project requires a mosaic engine that is portable, uses only standard astronomy packages, is highly scaleable and is easy to fine-tune. These are the goals of Montage, which is therefore a serious candidate for GLIMPSE processing.

## **Use Case IV – Outreach**

Large-scale image mosaics are useful in promoting general interest in infrared astronomy through their use in local image galleries as well as the development of posters, pamphlets, and other media for both the general public and educators. Mosaics showing data at multiple wavelengths on common projection, spatial scales etc exert a powerful influence on the imagination, especially when made part of a larger permanent display at a museum or planetarium. Access to Montage will allow production of large scale images from multiple data sets that would otherwise be very labor-intensive to accomplish.

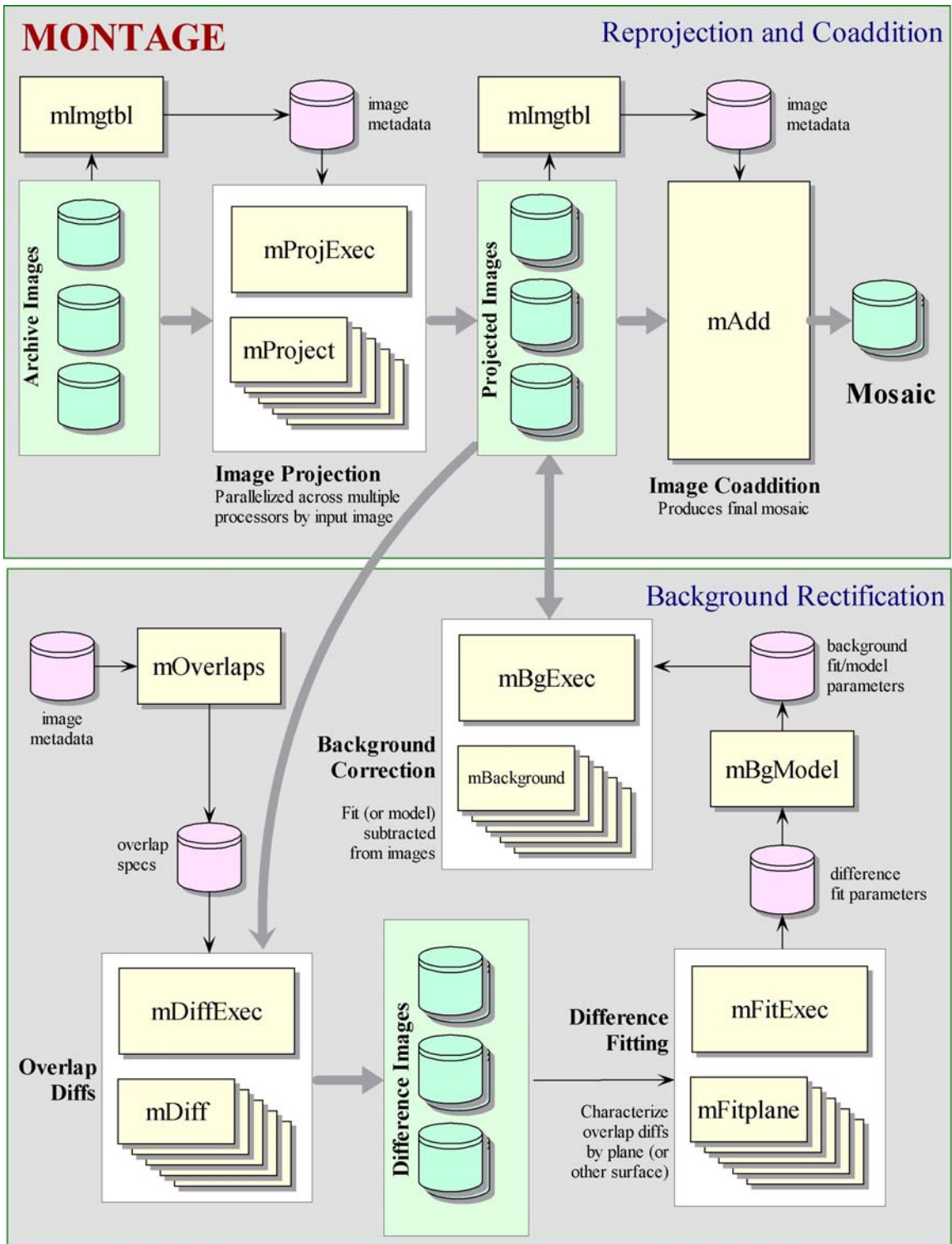
## **6 Design Elements of Montage**

The design elements of Montage that respond to the technical and science drivers described in Section 5 are summarized here. There are two major areas of development:

- A compute engine that performs all reprojection and coaddition of input images and that will generate the mosaics
- A background rectification engine that removes background and instrumental radiation from the images. This software will be generally be run independently of the mosaic engine to generate database of background parameters that the mosaic engine will then read to remove backgrounds.

Figure 1 identifies the components that we will design and build and the high level interfaces between them. Table 2 describes the components in more detail.

### **Figure 1: Design Components of Montage**



**Table 2: Description of the Design Components of Montage**

<b>Component</b>	<b>Description</b>
<i>Mosaic Engine Components</i>	
<b>mImgtbl</b>	Extracts the FITS header geometry information from a set of files and creates an ASCII image metadata table from it used by several of the other programs.
<b>mProject</b>	Reprojects a single image to the scale defined in a pseudo-FITS header template file (an ASCII file with the output image header lines, but not padded to 80 characters and with new lines at the end of each line). Actually produces a pair of images: the reprojected image and an "area" image consisting of the fraction input pixel sky area that went into each output pixel.
<b>mProjExec</b>	A simple executive which runs <code>mProject</code> for each image in an image metadata table.
<b>mAdd</b>	Coadd the reprojected images using the same FITS header template and working from the same <code>mImgtbl</code> list.
<i>Background Rectification Components</i>	
<b>mOverlaps</b>	Analyze an image metadata table to determine a list of overlapping images.
<b>mDiff</b>	Perform a simple image difference between a single pair of overlapping images. This is meant for use on reprojected images where the pixels already line up exactly.
<b>mDiffExec</b>	Run <code>mDiff</code> on all the pairs identified by <code>mOverlaps</code> .
<b>mFitplane</b>	Fit a plane (excluding outlier pixels) to an image. Meant for use on the difference images generated above.
<b>mFitExec</b>	Run <code>mFitplane</code> on all the <code>mOverlaps</code> pairs. Creates a table of image-to-image difference parameters.
<b>mBgModel</b>	Modeling/fitting program which uses the image-to-image



	difference parameter table to interactively determine a set of corrections to apply to each image to achieve a "best" global fit.
<b>mBackground</b>	Remove a background from a single image (planar has proven to be adequate for the images we have dealt with).
<b>mBgExec</b>	Run <code>mBackground</code> on all the images in the metadata table

## 7 Major Areas of Effort

There are five important technical and managerial issues that must be addressed in building Montage and deploying it within the NVO.

### 7.1 Interface with Astronomical Community: Science Goals

Maximal scientific return requires “buy-in” of the astronomical community in the development of science requirements, use cases and validation. Montage must also support the broad science-goals of the National Virtual Observatory.

### 7.2 Interface with 2MASS, DPOSS and SDSS Projects

Background removal algorithms are generally specific to a given project, and Montage must therefore leverage the expertise of the 2MASS, DPOSS and SDSS projects.

### 7.3 Deployment within the Architecture of the National Virtual Observatory

This service is intended as a cornerstone compute-intensive service of the NVO. Montage must be registered in a master catalog of NVO services, and must be tested in the NVO test bed before being considered an NVO service. The development of the NVO framework will take place concurrently with Montage development, and therefore the two efforts must be managed and scheduled so that they remain “in-synch”.

### 7.4 Development of Montage

Montage will consist of two components, a compute engine and a background rectification service, described below.

#### 7.4.1 Compute engine

The heart of the service will be a compute-engine that generates the mosaic pixel-by-pixel. While it will run operationally on the *TeraGrid*, a portable version will be made available for download. The *yourSky* code, the baseline for this project, must be made incrementally faster, more portable, more science-grade and must meet mandated performance specifications for each delivery.

### 7.4.2 Background removal service

Diffuse background emission from the atmosphere and instrumental features must be eliminated and their impact on the integrity of the data must be quantified. Cooperation with the 2MASS, SDSS and DPOSS projects is essential in implementing and documenting algorithms that preserve the scientific fidelity of the input images data.

### 7.5 Visualization of Image Mosaics

Mosaics will generally be too big for easy download and storage on a user's machine. Montage must therefore manage delivery of mosaics to the user, by enabling subsampling, subsetting or resampling of the mosaic.

## 8 Project Structure

### 8.1 Project Life Cycle Model

Montage will employ the Evolutionary Delivery lifecycle, a combination of staged delivery and evolutionary prototyping. The project has "hard" milestones that will provide incremental improvements in performance and portability. While these milestones are well suited to a staged delivery model, the project must accommodate requests for new features, such as new algorithms for performing co-additions, flagging of bad or anomalous pixels, and flagging or mapping of outlying data points. The delivery dates will not change to accommodate these requests for new features, but the dates as negotiated do accommodate sufficient "headroom" for incorporating new requirements.

The requirements satisfied by each delivery will be tracked through a Requirements Traceability Matrix, described in the Quality Assurance plan.

### 8.2 Project Responsibilities

Table 2 lists the responsibilities of the Montage staff. The team is small, has strong working relationships and a sense of common intellectual ownership of the project. The functions of the Montage staff are kept separate as necessary to prevent internal conflicts of interest – development, testing and QA are, for example, all performed by different people.

**Table 2: Responsibilities of the Montage team**

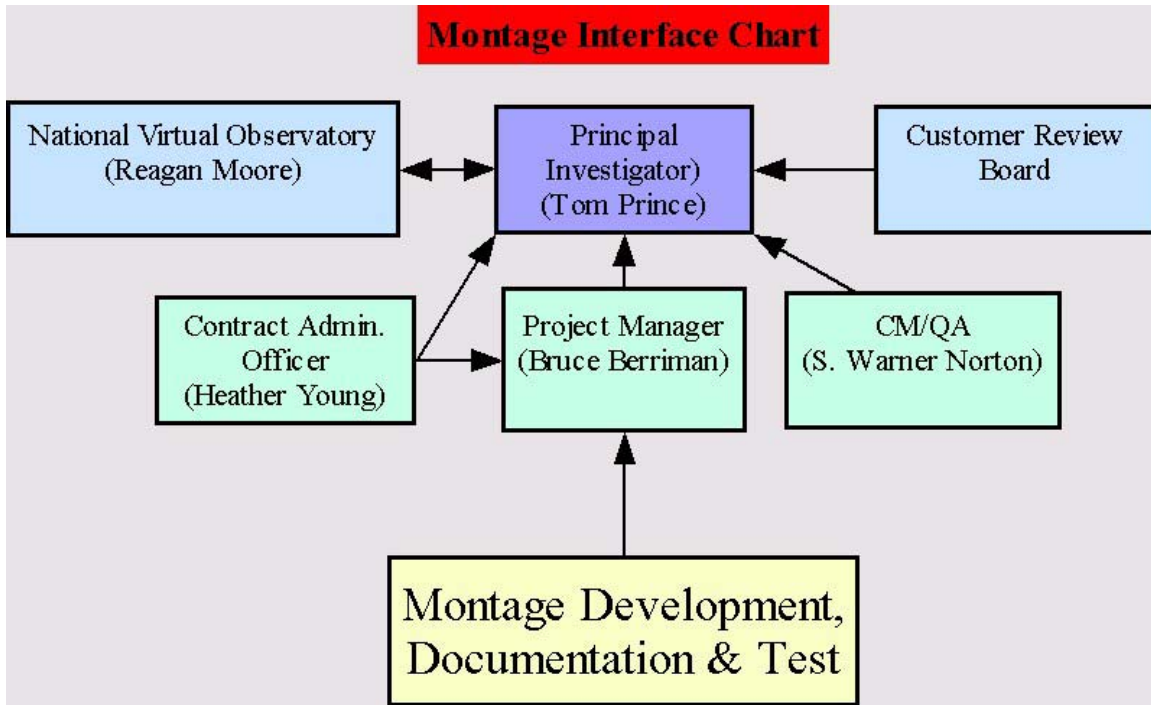
<b>Role</b>	<b>Responsibilities</b>	<b>Staff</b>
Principal Investigator	Overall technical leadership; external interfaces	T. Prince
Project Manager	Ensure team focus on milestones; budget planning; project planning; reporting to the PI	G. B. Berriman
Contract Administration Officer	Contract administration; budget reporting	H. Young

Risk Management	Identifying, tracking and mitigating risk	G. B. Berriman, J. Jacob
JPL Line Management	Interface with JPL for internal status reporting, finances and workforce	D. Katz
JPL Task Manager	Overall technical leadership of JPL effort	J. Jacob
Architecture	Overall design and development of MONTAGE, including compute engine & background removal algorithms	J. C. Good, J. Jacob, D. Curkendall
Configuration Manager	Install and manage CM system for code and documents	S. Warner Norton
Liaison with HPCC Community	Advise on compute resources available for testing	D. Curkendall
Quality Assurance	Ensure code quality & maintain all walkthrough materials	S. Warner Norton
Test and Validation	Design and perform test and validation plans	B. Berriman, D. S. Katz
Documentation	Prepare all documentation for public delivery	B. Berriman, D. S. Katz, J. Good, R. Williams, J. Jacob, T. Prince
Web page development	Design, deploy and maintain public and project web pages	S. Warner Norton

### 8.3 Project Interfaces

Figure 1 shows the internal (green) and external (blue) interfaces for Montage.

#### Figure 2: Montage Boundaries and Interfaces



### 8.3.1 Internal Interfaces

The internal interfaces are for reporting purposes:

- The Project Manager reports to the Principal Investigator on progress, technical and managerial issues and overall budget
- The CM/QA officer will report to the Principal Investigator on issues of software quality
- The contract administration officer will report budget issues and Caltech contract management issues to the Principal Investigator and the Project Manager

There is no formal reporting interface necessary between JPL and Montage project management. The JPL line manager will perform JPL internal reporting, and the JPL Task Manager will work as an integrated part of the technical team.

### 8.3.2 External Interfaces

The external interfaces are both technical interfaces: with the user community, to ensure maximal science value of the mosaic engine; with the NVO, to ensure that Montage is compliant with NVO requirements.

#### **Interface with the User Community: Customer Review Board**

The primary interface with the user community will be through the customer review board. Its charter is to advise the project on science requirements and algorithms for background removal, and to advise and assist in having users participate in validation of

Montage. The board will be staffed by practicing astronomers. A series of science requirements and science use cases recommended by the board is included as part of this plan. Thereafter the board will advise on modifications to the science use cases and additional functionality, and will be asked to give at least informal approval to software releases.

Membership:

Dr. Alex Szalay	(Johns Hopkins University)
Dr. Lisa Storrie-Lombardi	(SIRTF Science Center)
Dr. Thomas Jarrett	(Infrared Processing and Analysis Center)
Dr. George Djorgovski	(California Institute of Technology)
Dr. Michael Wolff	(University of Colorado)
Dr. Peter Martin	(Canadian Institute for Theoretical Astrophysics)

In addition, the project web site will solicit contributions from astronomers on suggested use cases and science requirements.

### **Interface with the National Virtual Observatory**

Dr. Reagan Moore is the Lead Architect of the National Virtual Observatory, and will act formally as a technical liaison between the NVO architecture team and the Principal Investigator. In addition, Drs. Berriman, Good and Williams are members of the architecture team and Dr. Williams is also the Co-Principal Investigator of the NVO. The Montage team members have a strong working relationship with Dr Moore and have worked with him to serve the 2MASS Atlas Image data set (10 TB) through the HPSS at SDSC.

There are three issues for resolution:

- Registration of Montage in a catalog of NVO services. Montage will be a test case for this registration mechanism, which will involve characterizing the service such that it can be discovered and run by third-party applications.
- Montage will run operationally on the *TeraGrid*, and so adequate processing power and storage space (file space and database) must be made available.
- IPAC is developing middleware that will manage and control jobs of indeterminate length, which will include jobs submitted to the .. Montage must ultimately interface to this middleware, but must also successfully function without it.

Solutions to these issues will be part of the NVO project rather than part of Montage, which will concentrate on the design and delivery of Montage as a fast interoperable service. A similar approach applies to other compute-intensive services for astronomy.

## **8.4 Management Approach**

### **8.4.1 Objectives, Dependencies and Constraints**

The principal managerial objective is to ensure that Montage is deployed on schedule and within budget, with all incremental deliveries made according to the schedule negotiated

with the sponsor. Fulfilling these obligations requires that the project adheres closely to industry-standards for software development.

The project makes the following assumptions:

- That the personnel are committed to the project for its three year duration.
- That the *TeraGrid* will be developed concurrently with the development of Montage.
- That the following computing resources are available throughout the project

Machine	OS	Status
<i>Operations:</i>		
<i>TeraGrid</i>	Red Hat Linux 6.2	Available as <i>TeraGrid Lite</i> . Request for time approved.
<i>Testing:</i>		
IBM Blue Horizon	AIX 5L	Account at SDSC approved
Linux Cluster	Red Hat Linux 6.2	Account at CACR approved
IPG SGI O2K, O3K	IRIX 6.5.x	Account approved for JPL
Solaris Workstations	Solaris 2.7, 2.8	Available at IPAC
Linux workstations	Red Hat Linux 6.2, 7.x	Available at IPAC, CACR
<i>Development:</i>		
Solaris workstations	Solaris 2.7, 2.8	Available at IPAC
Linux workstations	Red Hat Linux 6.2, 7.x	Available at IPAC, CACR

Montage requires access to Database Management Servers (DBMS's) that will house the background removal parameters, with one row for each image within a dataset. The TeraGrid will most likely make use of Oracle, already accessible by us through SDSC, and will be used in operations.

## 8.4.2 Monitoring and Controlling Mechanisms

### Staff Meeting Format

The Project Manager will host a face-to-face team meeting every two weeks, with more frequent meetings as needed. The biweekly meeting has three purposes:

- As a reporting session to discuss status
- As a working session to debate issues, and
- As a Configuration Control Board (CCB) session, since the team will act as its own CCB (per Configuration Management plan)

Meetings will have the following format:

- Report on tasks in progress and those finished
- New problems and issues
- Review of schedule and risks (per Risk Mitigation Strategy)
- Assign tasks as needed
- CCB business (if required)

- Extended discussion of issues

The project manager will keep the minutes, which will be published on the project internal website and distributed to the staff by email.

### **Reporting to PI**

The Project Manager will report progress and problems to the PI each week, either face-to-face or through short emails.

### **Reporting to Sponsor**

Following approval by the PI, the Project Manager will deliver annual reports to the project web site and the sponsor on August 31, 2002 and August 31, 2003 and will deliver the final project report to the web site on February 15, 2005.

## **9 Technical Approach**

### **9.1 Assumptions and Constraints**

Montage will be able to build small mosaics on a user's Linux laptop and be able to process many simultaneous, large requests on the *TeraGrid*. It must therefore be written to ensure portability, and make use of common tools and libraries that exist on all widely-used platforms.

### **9.2 Use of Open Source Software**

Montage will use a small set of standard open-source astronomical libraries for reading FITS image files, performing coordinate system transformations, and handling image projection/deprojection operations. These libraries are portable and well-tested, and a current version will be delivered with all releases of Montage (this version may be slightly out of date but can be replaced at the user's discretion). The libraries are:

<b>Library</b>	<b>Description</b>	<b>Current Release</b>	<b>Origin</b>
CFITSIO	FITS reader	Version 2.300	HEASARC
WCSTools	Image projection	Version 3.0.5	SAO
Coord	Coordinate transformation	Version 1.2	IRSA

### **9.3 Portability of Montage Software**

To ensure maximal portability and use of Montage, it will not use shared-memory, specific DBMS interfaces, or platform-specific libraries, and it will minimize its use of memory (as long as it does not compromise the quality and range of the algorithm). Ancillary information, such as tables of information on the collection of images that are being processed, will be captured in simple text files that can be parsed by any computer.

Montage will be constructed to operate entirely from command-line arguments, using the ancillary files describe above to communicate other information needed to process a request.

Montage will be developed in ANSI-standard C. It will be guaranteed to compile with GNU gcc and to build with GNU gmake. Other compilers and IDE's will almost certainly work just as well, though we make no guarantees about testing such and will only do so as resources permit.

Montage will be built on several UNIX platforms, including but not limited to Solaris, AIX, and Linux. It will of course be tested and run operationally on the TeraGrid. We will formally test Montage on IRIX only as resources permit.

## **9.4 Delivery of Maintainable Code**

Maintainable code will be ensured by developing modular code, with each module (or set of modules) dedicated to performing specific tasks needed in building a science grade mosaics (see section 4.2). The code will be developed according to the style guidelines in <http://www.construx.com/survivalguide/> → Coding Standard

## **9.5 Activities, Tools and Products**

Following internal agreement on design, we will define a set (or set of modules) that will perform the tasks necessary to build a science-grade image. These tasks will include:

- image reprojection/resampling,
- background fitting,
- image overlap analysis,
- local/global background correction, and
- image merging (weighted coaddition, pixel median filter coaddition, replacement coaddition).

This task list and the expected command-line arguments and ancillary files will be reviewed by the Customer Review Board.

We will then define the precise methodology to be used in pixel resampling and prototype this to determine the effects of computational round-off and any shortcomings of the resampling methodology.

Models for background estimation and correction for the 2MASS, SDSS and DPOSS datasets must be encoded. The Montage project will leverage existing background algorithms developed by these projects, rather than attempt to develop them internally.

## **9.6 Build Strategy**

The Montage project will quickly construct an end-to-end system in the first phase of the project. The aim will be to produce mosaics that are accurate and can be used by scientists as a reliable science tool from the very first release. Accurate in this context refers to accuracy in astrometry, uniformity in photometry/calibration and robust against image defects or bad data values. Subsequent phases, as outlined in the project plan, will



focus principally on increasing speed, robustness, and portability, per the detailed project milestones.

## **9.7 Maintenance Strategy**

Following each release, the team will correct as a top priority such defects as are reported by users that cause Montage to fail or give bad or incorrect output. Requests for enhanced functionality will be considered by the team. After each release, there will be a request period (whose length will be determined by the team) when requests for enhanced functionality will be entertained by the team, who will decide which requests to fulfill in subsequent releases, based on project resources and scientific value; the CRB will be polled on the latter. Enhancements will generally be released only as part of the release cycle given in the Milestones, except when an upgrade is necessary for a project to meet mission critical goals.

A new release will supercede the previous one, and therefore only the current release will be supported.

## **10 Risk Mitigation Strategy**

### **10.1 Definition of Risk**

Risk is defined as “The potential for undesirable outcomes which involve loss of time, money, product quality or functionality.”

### **10.2 Organizational Responsibilities**

Montage will operate on a strict budget and “hard” milestones, and special attention must be paid to those risks affecting budget or schedule and must be reported to the sponsor. Consequently, the responsibilities for Risk Management will be distributed as follows:

- The Project Manager is responsible for acting as Risk Mitigation officer for the project. He/She will be responsible for identifying and tracking all risks, developing risk mitigation strategies and overseeing them.
- The JPL Task Manager will act as Deputy Risk Mitigation Officer, and assist the Risk Mitigation Officer in identifying and tracking risks, and developing risk mitigation strategies.
- As part of his weekly reports, the Project Manager will inform the Principal Investigator of budget and schedule risks. The P.I. will approve all strategies for mitigating such risks, and will inform the sponsor of their impact on the project as necessary.
- The Montage team will be jointly responsible for developing strategies for minimizing technical risks.

### **10.3 Schedule for Baselineing and Updating Project Risk**

The results of an initial risk assessment will be documented on the Project Internal web page on July 31, 2002 (at the end of the design phase) and will be delivered to the

sponsor. The documentation will consist of a table that lists the following information: Description of Risk, Date Open, Mitigation Plan and Progress, Date Closed. The risks will be listed in order of priority. A sample (fictitious) risk is as follows:

Description	Date Open	Mitigation Plan/Progress	Date Closed
TeraGrid will not be available on required timescale for testing/ deployment	July 31, 2002	Get TeraGrid release schedule. Prototype TeraGrid already deployed. Figure out when we need to have it for testing purposes, and find out what we need to do to get time on it. Need a contingency plan – what alternative systems are available? Need to test on more than one system anyway to test portability. Develop a failover plan – needed anyway if TeraGrid is operational but down. Final milestone only requires TeraGrid – three years are available to us mitigate this risk. Actions: RW – get TeraGrid schedule; DK, DC – investigate and report on other grid resources available to us. Do this by 10/31, then begin failover development plan based on the intelligence we receive.	

Risks will subsequently be reviewed and extended and re-prioritized every two weeks at project meetings.

### 10.4 Methodology for Risk Identification & Mitigation Plans

A common strategy will be used during the initial risk assessment and for subsequent reviews and updates. Every two weeks at project meetings, the team will be responsible for:

- Proposing and assessing risks, and developing a risk mitigation plan for each risk.
- Assessing progress in mitigating existing risks
- Reviewing the prioritization of existing and new risks

### 10.5 Risk Tracking

After project meetings, the Project Manager will update documentation of existing risks, add new risks to the table, and re-order the risks according to changes in their priority. He will inform the PI of budget and schedule risks, as necessary, and will update the project schedule to take account of risk mitigation efforts.

## 11 Configuration Management

### 11.1 Items Subject to Configuration Control

The CM plan describes procedures for managing changes to the following items:

- **Project Documentation:**
  - Software engineering plan, including QA plan, risk mitigation plan, CM plan, technical and management approaches and project milestones (this document)
  - Software requirements
  - Science requirements and use cases
  - System design.
  - All user documentation
- **Public Web Site**

- **Database Schema** (required for background removal)
- **All software deliverables**, for the mosaic compute-engine and the background removal engine, including source code, build instructions, dependency files and README and other files that describe the software.

The CM plan also describes procedures for

- Delivery and maintenance of released versions of MONTAGE
- Backup and long-term storage of project deliverables.

## **11.2 Organizational Responsibilities**

Configuration Management will be the responsibility of Ms. Saille Warner Norton. She will report directly to the Principal Investigator, and her responsibilities as CM officer will be:

- Tracking change requests and their disposition
- Ensure that all approved change requests are documented & that the changes have been made as approved
- Ensuring that all changed materials (documentation or code) meet Montage standards, and are delivered and stored according to the procedures described below
- Maintain the CM tools, principally the defect tracker the CVS version control system.

## **11.3 Configuration Management Tools**

Documents will be edited with Microsoft Word, and other formats (html, pdf, text) will be derived from Word versions.

Database Schema will be documented in Microsoft Excel, and other formats (html, pdf, text) will be derived from the Excel version

Software revision control will be through the Concurrent Versions System (CVS).

Defect tracking and change requests will be made through a commercial or Open Source tool. Montage will choose and deploy such a tool by July 31, 2002.

### **11.3.1 Change Management Board**

Because Montage is a small project staffed locally, it can approve changes quickly without a formal Configuration Control Board (CCB) that would ordinarily represent the segments of a large or distributed project. The Montage team will in effect act as its own CCB, chaired by the Project Manager, and it will use the bi-weekly Staff Meetings to discuss the scope and legitimacy of changes, and approve changes to configured items. In case of dispute, the PI will make binding decisions.

### **11.3.2 Recording Change Requests**

Change requests for *all* configured items will be tracked through the defect tracking tool that is selected for this project. All approved changes will be recorded in Revision Histories, as described in subsequent sections.

## 11.4 Configuration Management Procedures for Documentation

### 11.4.1 Documentation Repository

All versions of configured project documents will be written in Word and be kept in a dedicated master directory, with read-only world access. These documents are:

- Software engineering plan, including QA plan, risk mitigation plan, CM plan, technical and management approaches and project milestones (this document)
- Software requirements
- Science requirements and use cases
- System design.
- All user documentation

### 11.4.2 Versioning for Documentation

Documents will use a two digit version number, X.Y, where

- X represents substantial changes in technical content (addition of new processes, revisions to existing processes, etc.)
- Y represents corrections and clarification to existing processes

### 11.4.3 Procedure for Recording Revision History

The second page of each document will record its full release history, in accord with this template (examples are fictitious)

#### MONTAGE SOFTWARE ENGINEERING PLAN REVISION HISTORY

Version	Description of Revision	Date
1.1	Edit QA Plan to add “create backup CD” to release checklist	July 31, 2002
1.0	Initial Release	May 31, 2002

## 11.5 Configuration Management Procedures for the Public Web Site

### 11.5.1 Web Page Source Repository

The source code for all revisions of the public site will be committed to CVS.

### 11.5.2 Versioning for the Public Web Page

The version number of the public web page need not be declared on the public site, but internally the source code will follow two-digit versioning, X. Y:

- X represents substantial changes in technical content (addition of new documents, image mosaic examples, etc), generally coinciding with the project milestones.
- Y represents corrections and clarification to existing content.

### 11.5.3 Procedure for Recording Revision History

The public page will contain a link to a tabulation of the history, in line with this template (examples are fictitious).

## MONTAGE WEB PAGE REVISION HISTORY

Description of Revision	Date
Deliver Design Policy for Interoperability	July 31, 2002
Initial Release	May 31, 2002

### 11.6 Configuration Management Procedures for Database Schema

The database schema will be documented in spreadsheets that describe the columns of the database tables. There will be separate spreadsheets for the 2MASS, DPOSS and SDSS datasets, and each spreadsheet will specify the appropriate background removal parameters; an associated data dictionary will describe the characteristics of each column.

#### 11.6.1 Database Schema Repository

Descriptions of the database schema (database and data dictionary) will be kept in the dedicated master directory used for documentation.

#### 11.6.2 Versioning for Documentation

The schema will use a two digit version number, X.Y, where

- X represents substantial changes in technical content (addition of new columns, new fields in the data dictionaries, etc.)
- Y represents corrections and clarification to the documentation of existing database tables.

#### 11.6.3 Procedure for Recording Revision History

Each spreadsheet will include a full release history, in line with this template (examples are fictitious).

### MONTAGE DATABASE SCHEMA REVISION HISTORY

Version	Description of Revision	Date
1.1	Column 'Parameter X' is now indexed	July 31, 2002
1.0	Initial Release	May 31, 2002

### 11.7 Configuration Management Procedures for Test Datasets

#### 11.7.1 Test Data Sets Repository

The Montage project will identify test data sets that will be delivered with the software as part of each code improvement or interoperability milestone. These test data sets will be in FITS format. Each version of configured test data sets will reside in a dedicated read-only directory. The directory will also contain documentation on the test data sets, including change history (see below), data files in the data set and their version numbers,

and a detailed description of the contents of each file, including but not limited to: project or mission from which the file has been drawn, region of sky, wavelengths covered, coordinates, projection, resolution and sampling, reasons why file is part of the test data set, and any modifications that have been made to the file as delivered by the project for the purpose of expediting testing. The same information will also be written to the headers of the FITS files.

### **11.7.2 Versioning for Test Data Sets**

Test data sets will have their own version numbers that are independent of the software version numbers. They will have a two digit version number X.Y, where:

- X represents addition, removal or replacement of files within a test data set
- Y represents changes to any given file in the data set, such as a correction to an incorrect file header, or a modification to support more efficient or higher quality testing.

Individual files will have three digit version numbers X.Y.Z, where X and Y are as above, and Z will increase by one if that file was modified when Y was increased. When a file is added to a test data set, Z will always be zero. The versions of individual files will also be written to the FITS headers.

### **11.7.3 Procedure for Recording Revision History**

Each new version of the test data sets will contain documentation in flat ASCII format that will contain a complete release history that is in line with the following format:

Version	Description	Date
1.1	Corrected record of projection in FITS headers	August 20, 2002
1.0	Initial Release	June 30, 2002

## **11.8 Configuration Management Procedures for Software**

### **11.8.1 Software Repository**

Montage will use the Concurrent Version System (CVS) to provide a repository for software modules, dependency lists, makefiles and associated ASCII documentation file, such as build instructions. The directory structure laid out under CVSROOT under CVS will be a mirror of the development and operations directory structure. Developers can check out files, edit them and commit changes.

### **11.8.2 Versioning for Software**

All modules and libraries will have version numbers, even in a development area. The version number should be increased, even in a development area, whenever changes are made. These changes will be committed to the revision control system, CVS.

Modules and libraries will use a three digit number X.Y.Z. where

- X represents major code improvements, such as new functionality
- Y represents defect corrections
- Z is reserved for “make install” runs (when dependent modules change but the application does not).

The Montage system releases will have the following version numbers, corresponding to code improvement project milestones:

Version	Milestone	Release Date
Version 1.0.0	F) First Code Improvement demonstrated	February 28, 2003
Version 2.0.0	G) Second Code Improvement	February 28, 2004
Version 3.0.0	K) Customer Delivery	January 10, 2005

The interoperability prototypes will not have separate version numbers because substantial new functionality is not planned for these releases.

### **Policy for Referencing Version Numbers in Services, Libraries and Directories**

- "Standard" libraries and executables on development, test and operations environments will be soft links pointing at the operations version. (e.g. FITSIO--> FITSIO\_v1.5.0). This convention ensures the most current release is used even if there are newer versions under test or development.
- All development versions of services and applications should be built with the Makefile pointing to these standard versions except in those instances where there is a dependency on new functionality or bug fixes only available in a newer version. In this case, the Makefile should point to the new version explicitly (e.g. -FITSIO --> -FITSIO\_v1.6.0) and the Dependencies file updated.
- Version numbers on libraries and service executables should be incorporated into the name in a consistent manner (e.g. libsvcio\_v2.1.4.a). Version numbers on applications directory names should be incorporated into the name in the same consistent manner (e.g. SurveyVis\_v3.2.1).

### **Dependency Tracking**

CVS provides version control, but offers no capability for tracking changes in dependent software. Montage has very few dependencies – mainly on Open Source libraries, which change infrequently. Dependency tracking will therefore be performed manually.

Briefly, dependencies for each release will be recorded in simple text files that record, in a uniform way, the name of the dependency, its version number and, if known, the release date. This file will be committed to CVS along with the software modules.

### **Procedure for Documenting Revision History for Software**

The changes made to a module will be entered by the developer whenever they commit a new or updated module to CVS.

For releases of the Montage software, a release history will be posted on the web page when the software is made available for download.

### **11.8.3 Delivery and Maintenance of Released Versions of Montage**

Software will be delivered following successful completion of all items on the release checklist:

<http://www.construx.com/survivalguide/> -> Release Checklist

This list includes insuring all testing is complete, all defects have been corrected etc.

### **11.8.4 Backup & Restore and Long-Term Storage of Project Deliverables**

#### **Backup & Restore**

All documents, source code and test data sets will be backed up as part of IPAC's routine daily backups, and restored using standard IPAC restore procedures. Restores will be performed quarterly to ensure that the system can be rebuilt from scratch.

#### **Long-term Storage**

- For each release of Montage, copies of released documentation, database schema, web page code and source code will be backed up on two CDs, one of which will reside at IPAC, and the second will reside outside the Caltech campus.
- Hard copies of released documentation, database schema, web page code and source code will be kept at IPAC / Caltech.
- All review materials will be available on the project internal web site and copies will also be kept on CD-R.

## **12 Quality Assurance**

This chapter describes the Quality Assurance activities that will be performed on the Montage software system. The quality assurance plan must ensure that each release of the Montage software is robust, accurate, maintainable and portable, and satisfies its requirements.

These goals will be met by three main activities:

- Technical reviews of software requirements, design and code
- Development and execution of test plans



- Monitoring performance of released software.

The QA activities will be supported through a Requirements Traceability matrix, a defect tracking tool, and collection of project metrics.

## 12.1 Organizational Responsibilities

The QA team will consist of personnel who are uninvolved in development efforts.

- Ms. Saille Warner Norton will act as Quality Assurance and Configuration Manager. She performs this task for the IRSA, and the QA responsibilities for MONTAGE will simply be an extension of her IRSA responsibilities. She will organize and moderate technical reviews, take part in the testing, keep track of defect reports and ensure that all QA activities are documented before each release of MONTAGE
- The Project Manager and JPL Task Manager will be jointly responsible for development of all test plans, and will lead the execution of the test plan. They will also be responsible for engaging users in testing and validation.
- The MONTAGE team members will be responsible for correcting and documenting defects identified as part of QA activities.

## 12.2 Quality Assurance Activities

### 12.2.1 Technical Reviews of Software Requirements, Design and Code

For each release of Montage (baseline delivery excepted), we will hold technical reviews of requirements, design, implementation, testing, and release. Potential customers will be invited to participate.

The reviews will focus on:

- Detecting defects and ensuring that coding standards are met
- Posting on the internal project web site reports that list defects and provides a schedule for correcting them
- Entering defects into the defect tracing system
- Ensuring that work identified in the reviews is performed

Defects reported in reviews will be tracked through the defect tracking tool.

### 12.2.2 Development and execution of test and validation plans

The following types of testing will be performed; a separate test plan will be written for each set of tests, with the preliminary test plan due July 31, 2002. Defects reported during testing will be tracked with the defect tracking system. As part of the test and validation effort staff, the Montage staff will work with the ESTO-CT contact scientists to instrument the code so that performance milestones can be measured in a meaningful way

Test Process	Description and Value
Developer Testing	Testing of individual modules by developers

Functional Testing	Testing of individual functions according to whether they satisfy entry and exit criteria
Integration testing	Exposes faults between interfaces
System testing	Ensures that requirements have been satisfied
Regression testing	Retest code following modification to ensure that new defects have not been introduced
Installation testing	Determines that users can download, build and use MONTAGE
Stress testing	Ensures that MONTAGE will run at or beyond the limits of its specifications
End User Manual Testing	Document is complete, consistent and error-free.
Beta Testing	Arrange for testing of MONTAGE at a site not involved in testing
Acceptance Testing	Determine that MONTAGE satisfies defined user criteria.

### 12.2.3 Test Data Sets

Montage will identify test data sets that will be the backbone of the test efforts. These data sets will be subject to configuration control and delivered with the code as part of each code improvement and interoperability delivery. The test data sets will be subsets of the public 2MASS, DPOSS and SDSS image data sets. With the permission of the releasing project, some of the original files may be modified to trap exceptions and anomalies. Such changes will allow for thorough testing of Montage with a test data set that is reasonably compact in size.

### 12.2.4 Monitoring the Performance of Released Software

Users will report defects and request upgrades through the defect tracking tool, which will be accessible through the public web page and will send defect reports to the cognizant developer. Open defects and enhancement requests will normally be reviewed at each staff meeting, but in cases where an emergency fix appears necessary before the next meeting, the developer will contact other members of the team to decide on the schedule. Defects that clearly degrade the performance of the software or cause it to fail will be corrected as part of routine maintenance. Requests for enhancements will generally be considered for the next release, but may be made sooner if there is clear scientific need. The Montage staff will seek guidance on such issues from the customer review board.

## 12.3 Quality Assurance Tools

### 12.3.1 Requirements Traceability Matrix

This matrix will be posted on the project internal web site and will be used to ensure full requirements coverage. The matrix will be an Excel spreadsheet table with column entries as below, with one entry for each requirement with a (fictitious) sample entry

Req. Spec. No.	Req. Statement	S/W module	Test Spec.	Test Case #	Verification	Mod. Field
10	Support coadditions through simple averaging and medians	Do_co_adds.c	Developer,functional, integration, acceptance, beta	16,19-22, 34,66-69	Fully verified	Version 2.0 requirements review – added median co-adds
etc						
etc						

The interpretation of the columns is as follows:

<b>Column</b>	<b>Description</b>	<b>Timeline</b>
Requirements Specifications Number	The requirement paragraph number as listed in the Requirements Specification document	initial requirements analysis
Requirement Statement	Paraphrase of the actual requirement as it appears in the Requirements Specification document	initial requirements analysis
Software Module	The module/subroutine that addresses the requirement	Detailed design phase
Test Spec.	The Test plan that contains the test case/procedure that validates the requirement. e.g., unit test plan, Integration test plan, Acceptance test plan	Test Plan development
Test Case #	The test cases that will be run to verify the requirement	Test Plan development
Verification	How well the requirement was verified: not verified; partially verified; fully verified	after executing the test procedure
Modification Field	Used in case requirement has been modified in any way throughout the life of the project. Indicate disposition (changed / eliminated / replaced), and authority for modification, e.g., eliminated – Requirements review, 10/17/01	Throughout the project as requirements are modified

### 12.3.2 Defect Tracking

The project will deploy a defect tracking system to records and track defects reported in reviews, testing and reported by defects. Defect tracking will take place throughout the project, beginning with initial design review. The defect tracker will report the following information:

- Identifier
- Description
- Priority
- Montage version number
- Development Phase (requirements, design, ..., user reported)
- Status (Either open or closed)
- Reporter
- Platform (OS, CPU, etc.)
- Submission Date
- Close-Out Date
- Resolution
- Staff Hours to Resolve Defect

### **12.3.3 Project Metrics**

The Project Manager and QA manager will be responsible for collecting the following metrics for all software releases (baseline release excepted). Separate metrics will be kept for the compute-engine (grid and portable versions) and the background server. All but the time-accounting metrics will be published on the web site, as they are of value to users.

### **12.3.4 Time-accounting metrics**

- For each release, level of effort broken down as follows for the mosaic-engine and for the background server code (actual vs. planned):
  - Design
  - Development
  - Testing
  - Validation
  - Maintenance (as applicable)
  - Documentation
  - Requirements Definition
  - Project Management

### **12.3.5 Software Size and Complexity**

- Count of requirements (number of “shalls”)
- Number of modules
- Number of source lines of code
- Number of Test Cases Planned and Executed
- Changes in the above over all releases (except for first code improvement)

### **12.3.6 Database Size <sup>†</sup>**

- Number of rows in database for each dataset
- Number of parameters describing background parameters for each dataset
- Changes in the above over all releases (except for first code improvement)

<sup>†</sup> Background server only

### **12.3.7 Software Performance**

- For each delivery, report for all machines on which Montage has been tested:
  - Throughput (square degrees processed per minute)
  - Scaling curves
  - Sustained bandwidth to disk

### **12.3.8 Software Complexity**

- McCabe metric or equivalent for each delivered module
- Changes in this metric for successive releases (except for first code improvement)

### **12.3.9 Schedule**

- Number of calendar months to meet each internal and release milestone (actual vs. planned)

### **12.3.10 Software Quality**

- Number of defects reported per release in:
  - Reviews
  - Testing
  - Operations (reported by users)
  - Level of effort to correct each defect
  - Length of time the defect was “open”
- Number of defects, ordered by priority, open as function of time.

## 13 References

[1] “Towards the National Virtual Observatory: A Report Prepared by the National Virtual Observatory Science Definition Team.” <http://nvosdt.org/sdt-final.pdf>

## 14 Acronyms

<b>2MASS</b>	Two Micron All Sky Survey
<b>AIX</b>	Advanced Interaction EXecutive
<b>ANSI</b>	American National Standards Institute
<b>API</b>	Applications Programming Interface
<b>ASCII</b>	American Standard Code for Information Interchange
<b>CCB</b>	Configuration Change Board
<b>CD-R</b>	Compact Disk –Read only
<b>CM</b>	Configuration Management
<b>CPU</b>	Central Processing Unit
<b>CRB</b>	Customer Review Board
<b>CVS</b>	Concurrent Versions System
<b>DBMS</b>	DataBase Management System
<b>DPOSS</b>	Digital Palomar Observatory Sky Survey
<b>ESTO-CT</b>	Earth Sciences Technology Office- Computational Technologies
<b>FITS</b>	Flexible Image Transport System
<b>GB</b>	Giga Byte
<b>Gflop</b>	Giga Floating Point Operation
<b>GLIMPSE</b>	Galactic Legacy Infrared Midplane Survey Extraordinaire
<b>GNU</b>	Gnu’s Not Unix
<b>HEASARC</b>	High Energy Astrophysics Science ARChive
<b>HPSS</b>	High Performance Storage System
<b>IBM</b>	International Business Machines
<b>IDE</b>	Interactive Development Environment
<b>IPAC</b>	Infrared Processing and Analysis Center
<b>IPG</b>	Information Power Grid
<b>IRSA</b>	InfraRed Science Archive
<b>IRAC</b>	InfraRed Array Camera
<b>JPL</b>	Jet Propulsion Laboratory

<b>MIPS (I)</b>	Machine Instructions per Second
<b>MIPS (II)</b>	Multiband Infrared Photometer for SIRTf
<b>NPACI</b>	National Partnership for Advanced Computing Infrastructure
<b>NVO</b>	National Virtual Observatory
<b>NVSS</b>	NRAO VLA Sky Survey
<b>OASIS</b>	On-Line Archive Science Information Services
<b>OS</b>	Operating System
<b>QA</b>	Quality Assurance
<b>ROSAT</b>	ROentgen SATellite
<b>SAO</b>	Smithsonian Astrophysical Observatory
<b>SDSC</b>	San Diego Supercomputer Center
<b>SDSS</b>	Sloan Digital Sky Survey
<b>SIRTf</b>	Space Infrared Telescope Facility
<b>SRB</b>	Storage Resource Broker
<b>VLA</b>	Very Large Array
<b>WCS</b>	World Coordinate System



