Introduction and Goals

The LSST Data Management System (DMS) will process, store, and retrieve vast quantities of data and metadata using large quantities of hardware and many instances of software configured into systems spanning continents. In order to meet the telescope's overall science requirements, the DMS must continue to operate even when some of these components have failed. This document describes the plan for achieving the necessary levels of fault tolerance for the overall DMS.

The DMS design must meet two key goals and one important constraint. First, it must preserve all raw images, calibration data, metadata, and associations between these items that are received from the telescope and its related instruments. This information is the fundamental scientific product and must be treated as the "crown jewels" of the project.

Second, the design must ensure that the system runs as intended by its developers in order to meet requirements for availability of the data. The stringency of these requirements differs substantially across the various subsystems within the DMS. The most stringent are the alert publication requirements. Catalog query and nightly archive processing requirements will be less rigorous, and data release processing and *ad hoc* science pipeline processing requirements are likely to be the least demanding.

The primary constraint is cost. The fault tolerance design must do as much as possible to achieve the above goals within budgetary limits for equipment, system development, and operations. Automation will be required to keep the cost of operating personnel low. This constraint also means that certain failure scenarios that inherently involve large recovery costs or have a very low probability of occurrence will not be planned out in detail and may not be addressed at all.

This document continues by laying out the multi-dimensional availability requirements for the various subsystems, whether derived from the science and functional requirements or implied by other system expectations. It describes the interfaces between the hardware infrastructure, middleware, application software, and science data quality analysis (SDQA) designs and teams with regard to fault tolerance. It then gives examples of expected fault types and specific use cases showing how faults may occur. A set of design patterns, comprising detection strategies, response strategies, and software frameworks implementing those strategies, is then described. The document concludes with a description of how subsystem developers will apply these design patterns, customizing them for unique situations.