Uptime and Reliability Analysis for NGAO

KECK ADAPTIVE OPTICS NOTE 684

Christopher Neyman
September 28, 2009

ABSTRACT

This note considers the initial uptime allocation for the NGAO and its top-level subsystems. The note also details the background needed to understand the equations used in the NGAO uptime budget and reliability analysis.

Revision History

Revision	Date	Author (s)	Reason for revision / remarks
1.0	September 28, 2009	Chris Neyman	Initial release

1. Introduction

Even at the best astronomical sites, periods of poor weather and atmospheric conditions place a priority on achieving astronomical observation in an efficient manner. Like most large telescopes, Keck Observatory has a high demand for observing time resulting in more proposals than can be accommodated each year. Astronomical sources can be observed for limited parts of the night and only during certain times of the year. Some events are transitory in nature. For these reasons, the NGAO must be a very reliable system. This note is a first step in "building in" the required reliability for the NGAO system from the start.

1.1. System Requirements

The NGAO System Requirements Document [1] in section 13.3 states that, "System downtime should be minimized by a combination of component reliability, ease of repair, maintenance, and appropriate sparing". The same section of the System Requirements Document (SRD), see Table 39 SRD v1.20, requires "less that or equal 5% of observing time should be lost to faults." This includes, "any loss to an exposure in progress and the time to start the next exposure after recovering from a fault." It is also required that the median time between faults or failures (MTBF) must be greater than or equal to 4 hours. Frequent short duration faults are not acceptable since they have a high impact on science productivity.

Table 24 of the System Requirements Document also states that, "The NGAO system must be able to support 200 night of science observing in a year". Assuming that the length of a night averaged over the course of a year is 12 hours and that morning and evening twilight are each on average an hour long results in a complete "night" of science observation being 10 hours. Therefore, the 5% downtime requirement implies that only 30 minutes can be lost to faults per night and that only 100 hours can be lost over the course of a year, which is 2000 hours of operation. As presently written, it is not made explicit whether these requirements apply to the NGAO system and possible science instruments or to just NGAO exclusive of science instruments. For the purposes of availability and reliability, estimates made in this document assume that the science instruments are included in the requirements and, as such, an availability allocation is made to them.

1.2. Scope

Although the NGAO work breakdown and the NGAO plan uses the term "uptime", in standard engineering practice the term availability is more accepted, and is used in the remainder of this document. The availability of a module or subsystem is the percentage of time when the system is operational. Downtime is simply the difference between availability and 100 percent. Downtime is often a more useful statistic to state when systems have a long time between failures.

The methods outlined in this note will be used to develop preliminary estimates of NGAO system availability. The analysis will be summarized in the availability or "uptime" budget for NGAO. This budget is analogous to the wavefront error budgets and will be used to partition the high-level availability requirement to a set of requirements on the NGAO subsystems. It can also be used to perform tradeoffs between subsystems. In addition, it can be used to allocate requirements between reliability and maintenance strategies. Finally, the budget should be compared to bottoms-up reliability estimates made by the designers of various subsystems.

The downtime requirement is allocated among approximately 20 NGAO subsystems. Downtime is the product of the failure rate and the time to repair a system after failure. The reliability analysis at this preliminary level does not dictate the tradeoff with the design of each subsystem between using high quality components, the number of spares, and the scope of a proposed maintenance plan. The component choices will affect the construction costs. The maintenance plan will affect the cost to operate NGAO. That tradeoff will be made as part of the design process of each subsystem. The following metrics for NGAO will interact with each other as the system engineering tasks proceed for the project:

- Construction Costs
- Operations Costs
- Downtime
- Performance Budget

The availability budget is made assuming that the system has reached a steady state after commissioning in which early failures (infant mortality) of components have been fixed and maintenance avoids faults as components reach their end of service. Both hardware and software reliability are considered when making availability allocation for each subsystem.

2. NGAO Subsystem Decomposition

The top level Product Breakdown Structure (PBS) used for the downtime analysis of NGAO is shown in Table 1.

NGAO Main Systems	Subsystems
AO System	AO Enclosure AO Optical Mechanical Controls Real-Time Controller
Laser Guide Star Facility	Laser Enclosure Laser System Beam Transport System LGS Control System Laser Traffic Control System LGS Facility Safety Systems
Science Operations Tools	Multi-System Command Sequencer Observing Tools User Interface Pre-Observing Tools Post-Observing Tools
Data Server	Data Server HW Data Server SW
MK Atmospheric Profile	
Science Instruments	NIR Imager NIR IFS

Table 1: Listing of the main systems and subsystems of NGAO used in the availability allocations.

3. Methodology for Calculating System Downtime and Availability

Most of the following sections are taken from references 2-4. The information is reproduced and discussed as it relates to the availability calculations made for NGAO.

3.1. System Failure Block Diagram

In order to make a calculation of the availability of a complicated system, a model of the interconnections of the systems parts must be produced. Such a model can be represented as a diagram or graph showing each subsystem as a box and the interconnection between subsystems represented as lines linking the boxes. An example of such a graph is shown in Figure 1. It shows the reliability model for the full Keck Observatory as it is setup for current LGS AO observations, not for NGAO. Any path leading from the left side of Figure 1, to the right side represents a success, as some science data will be recorded. A place in the model that has only one path between systems represents a single point failure in the system. For example, failure of the telescope drive motor would be represented as a break in the connections between the telescope block and all the blocks that follow. It is impossible to record data in this case. Redundancy is represented as parallel paths to success. A seeing limited instrument, typically NIRSPEC or DEMIMOS, is placed on the Nasmyth focal position opposite AO ready to be used in case of an AO failure. Therefore, even if the AO system fails the night is

not lost to science as a seeing limited observation is possible. In a similar manner, all LGS observing proposals were required to provide targets that could be done with natural guide stars (NGS). This option provides a path that allows AO observation to proceed with a laser system failure. Two AO science instruments provide a backup should one have a serious fault. The analysis that we preformed assumes that the weather is suitable in all cases. In a later section of this document, we will provide similar diagrams for the NGAO system. The next sections discuss how to calculate availability values from the block diagrams.

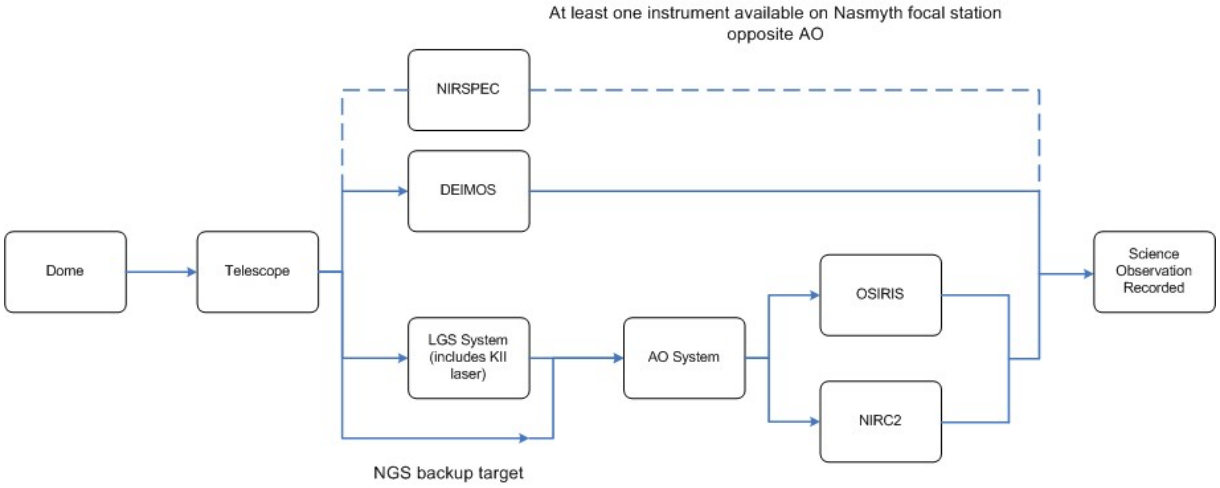

Figure 1: Current availability model for the entire Keck Observatory during LGS AO scheduled nights. It is possible to make science observations using LGS AO, NGS AO, or with the seeing limited instrument located at the Nasmyth focus opposite the AO enclosure.

3.2. Hardware Component Failure Model

Individual hardware components of a system will fail with a rate that varies with time as shown in Figure 2. The chance of a hardware failure is high during the initial life of the module. The failure rate during the rated useful life of the product is low. Once the end of the life is reached the failure rate of a module increases. This curve is named a "bathtub curve" after its characteristic shape. In the initial phases of component deployment, failures will occur because of design flaws and "infant mortality". Design failures take place due to inherent design flaws in the system. In a well-designed system, this class of failures should make a very small contribution to the total number of failures.

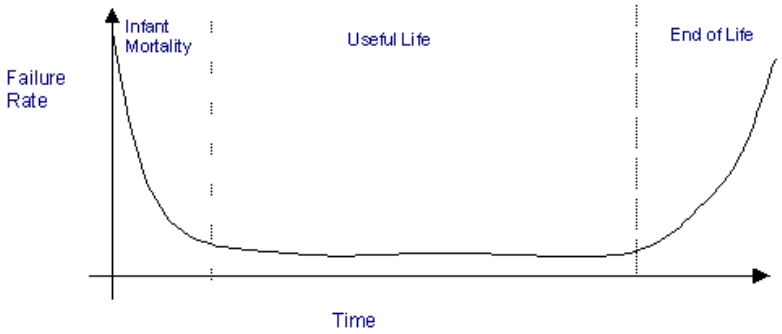

Figure 2: Three phase of most hardware components and systems.

Infant mortality failures are newly manufactured hardware faults. This type of failure can usually be attributed to manufacturing problems. These failures should not be present in NGAO subsystems as these faults will show up in factory burn in tests or in subsystem acceptance testing. Random failures can occur during the entire life of a hardware module. Once a hardware module has reached the end of its useful life, degradation of component characteristics will cause hardware modules to fail at a higher rate. This type of fault can be avoided by preventive hardware maintenance.

For the NGAO reliability models in this document, we assume that the system has reached steady-state failure typical of the useful lifetime section of the bathtub curve. The typical useful life of a unit must also be tracked during the design process as this will drive the cost of system maintenance.

3.3. Software Failure Model

Unlike the bathtub curve for hardware failures, software defects tend to have a declining frequency with years of operation. The failure rate of large and complex software asymptotically approaches zero and resembles the "infant mortality" or "burn-in" phase of the hardware bathtub curve throughout much of its life.

3.4. Mean Time Between Failures

Mean Time Between Failures (MTBF), as the name suggests, is the average time between failures of a subsystem. It is the average time before a failure occurs after repair or replacement of a system. A closely related term is the failure rate; it is the reciprocal of the MTBF and is given the symbol λ. For a system composed of N subsystems having no redundant components each with constant failure rate λ_i, the total failure rate of the system as a whole is:

$$Failure\,Rate = \sum_{i=1}^{N} \lambda_i \tag{1}$$

Combining equation (1) with the definitions of failure rate and MTBF results in the following relation:

$$MTBF = \frac{1}{Failure\,Rate} = \frac{1}{\sum_{i=1}^{N} \lambda_i} \tag{2}$$

The effective MTBF for a system and the $MTBF_i$ for each component are related by:

$$\frac{1}{MTBF} = \sum_{i=1}^{N} \frac{1}{MTBF_i} \tag{3}$$

MTBF for purchased hardware modules can sometimes be obtained from the vendor. MTBF for in-house developed hardware is calculated by the hardware team developing the unit, or by life testing multiple units until failure. MTBF for a complicated subsystem can be estimated by using vendor supplied component level data for MTBF and equations 1-3 given above to calculate the total MTBF. Because of equation 3, the MTBF will be dominated by the MTBF of the least reliable component. This method is typically used in the electronics industry to estimate the MTBF at the leave of electronic circuit boards from the MTBF data for the individual integrated circuits and other components on the board.

It is usually not practical to build and test to failure sample pieces of software. Instead, the following approach is often taken with software to characterize its failure modes.

Software failures can be characterized by keeping track of software defect density in the system. This number can be obtained by keeping track of historical software defect history. Defect density will depend on the following factors:

- Software process used to develop the design and code (use of peer level design/code reviews, unit testing)
- Complexity of the software
- Size of the software
- Experience of the team developing the software
- Percentage of code reused from a previous stable project
- Rigor and depth of testing before product is shipped

Defect density is typically measured in number of defects per thousand lines of code (defects/KLOC). MTBF for software can be estimated by simply multiplying the defect rate with KLOCs executed per second.

A final point about MTBF and the average lifetime of a single component before it fails in the field. These numbers are not identical. There exists a subtle statistical difference between them. MTBF and associated failure rate is a population average over a large number of identically manufactured or assembled components. This is a statistical ensemble average. The average lifetime is a time average before failure of a single component; this is a statistical time average. In general, these averages are not the same, i.e. the failure processes are not ergodic. An example taken from reference [3] may make the distinction clearer.

> A "thirty-something" American (well within his constant failure rate phase) has a failure (death) rate of about 1.1 deaths per 1000 person-years and, therefore, has an MTBF of 900 years (of course its really 900 person-years per death). Even the best ones, however, wear out long before that.

For NGAO, it is appropriate to use MTBF for reliability and downtime analysis. However, the maintenance schedule and spares decision should use the average lifetime of the relevant component or subsystem.

3.5. Calculating Mean Time To Repair

Mean Time To Repair (MTTR) is the time taken to repair a failed hardware module. In an operational system, repair generally means replacing the hardware module with an onsite spare. Thus, hardware MTTR could be viewed as mean time to replace a failed hardware module. It should be a goal of system designers to allow for a high MTTR value and still achieve the system reliability goals. A low MTTR requirement must be traded off against higher operational costs and the cost of purchasing a larger spares inventory. Some sample MTTR estimates for various servicing strategies from reference [4] are included in Table 2.

Estimating the Hardware MTTR		
Where are hardware spares kept?	**How is site manned?**	**Estimated MTTR**
Onsite	24 hours a day	30 minutes
Onsite	Staff is on call 24 hours a day	2 hours
Onsite	Regular working hours on week days as well as weekends and holidays	14 hours
Onsite	Regular working hours on week days only	1-2 days
Offsite: Shipped by courier when fault condition is encountered	Operator paged by system when a fault is detected.	1 week
Offsite: Maintained in a Keck controlled warehouse	System is remotely located. Staff needs to be flown in to replace the hardware.	2 week

Table 2: Some example service strategies and associated MTTR for hardware systems.

MTTR for a software module can be computed as the time taken to reboot after a software fault is detected. Thus, software MTTR could be viewed as the mean time to reboot after a software fault has been detected. This view is appropriate for application crashes that can be recovered by reloading the application or rebooting the system. However, it does not take into account software bugs, i.e. when real code changes must be made. In addition, many "software" problems are really related to hardware failures, so distinguishing the two will extend MTTR, as additional analysis and troubleshooting is required.

The goal of software designers should be to keep the software MTTR as low as possible. MTTR for software depends on several factors:

- Software fault tolerance techniques used
- OS selected (does the OS allow independent application reboot?)
- Code image downloading techniques

Some example software MTTR values and associated recovery methods from reference [4] are given in Table 3.

Estimating Software MTTR		
Software fault recovery mechanism	**Software reboot mechanism on fault detection**	**Estimate MTTR**
Software failure is detected by watchdog and/or health messages	Processor automatically reboots from a ROM resident image	30 seconds
Software failure is detected by watchdog and/or health messages	Processor automatically restarts the offending tasks, without needing an operating system reboot	30 seconds
Software failure is detected by watchdog and/or health messages	Processor automatically reboots and the operating system reboots from disk image and restarts applications	3 minutes
Software failure is detected by watchdog and/or health messages	Processor automatically reboots and the operating system and application images have to be download from another machine	10 minutes
Software failure detection is not supported or real code changes are required	Manually operator reboot is required	30 minutes to 2 weeks (software MTTR is same as hardware MTTR)

Table 3: Some example service strategies and associated MTTR for software systems.

In the rest of this document and the availability analysis that follows, we have adopted a special meaning for MTTR appropriate to an astronomical observatory. It is defined as the amount of science observations lost during nighttime hours before the system is back online and observing a science object. This is consistent with the system requirements [1]. After a system failure downtime stops accumulating at dawn or sooner if service is restored during the night, for example by rebooting a computer. In some cases, repairs cannot be done until personnel arrive next day. As long as the repair is complete by the next night, then total downtime does not accumulate further hours. Under these assumptions, on average a serious failure occurs half way through the night and costs 5 hours downtime before the system is back in operation the next night.

3.6. Availability and Downtime

Availability is the percentage of time that the system is operational and able to perform science observations. It can be calculated from the MTBF and MTTR as:

$$A = \frac{MTBF}{MTBF + MTTR} \tag{4}$$

The availability is the "probability" of a success observation. A consequence of equation (4) is that both MTBF and MTTR must be known or estimated to calculate the system availability. Since the system is either operational or not the downtime can be calculated from the availability, A, as:

$$D = 100\% - A \tag{5}$$

Downtime is sometimes stated in hours by multiplying equation (5) by the scheduled hours for the system.

The following table shows the implication for various availability percentages and the amount of downtime assuming the NGAO system is operational for 200 nights per year each of 10 hours long.

Availability	Downtime
90%	20 nights/year
95%	10 nights/year
99%	2 nights/year
99.9%	2 hours/year
99.99%	12 minutes/year
99.999%	1.2 minutes/year

Table 4: Availability and associated time lost assuming NGAO is operational for 200 nights a year and 10 hours per night.

3.7. Calculating Availability for Complex Systems

As noted in section 3.1, the reliability of a complex system can be understood by a block diagram that describes the redundancy and failure modes of a system. Typically, these diagrams can be analyzed by breaking them down into smaller diagrams where systems are connected in either series or parallel. Basic rules for these simple combinations can be used to determine the effective reliability of those pieces of the system. Then the rules can be used on the resulting blocks in turn. The analysis technique is similar to the calculation of the effective resistance of a complicated electrical circuit.

3.7.1. Availability for Systems in Series

A system made of N systems all connected in series is vulnerable to "a single point failure" as the loss of any system in the chain will take the full system down. The availability of such a system is calculated by,

$$A_{system} = A_1 A_2 \cdots A_N \qquad (6)$$

The effective MTBF for such a system can be calculated by equation (3) and the MTBF for each subsystem. The definition of availability in equation (4) along with the results of equation (6) can be used to calculate an effective MTTR for the combined system.

3.7.2. Two Systems in Parallel

A system that is made of two systems connected in parallel is considered to be operational if at least one or both systems are operational. The availability in this case is sum of the availability for both systems being operational plus the sum of each system not being operational while the other system is operational. The system availability in this case is

$$A_{system} = A_1 A_2 + A_1(1 - A_2) + (1 - A_1) A_2 \qquad (7)$$

In this case, the MTBF of the system is the product of the MTBF of each component subsystem. An effective MTTR can be calculated for the system from the definition of availability, equation (4), and the effective system MTBF.

3.7.3. Multiple Identical Systems in Parallel

Often systems are composed of several identical subsystems in parallel and only a fraction of them must be working for the system to operational. An example would be if NGAO were to have extra laser units in the system. We might have four lasers but require only three of them to be operational to consider the full system up. In general, for k subsystems in parallel, each with identical availability A_s, where we need at least r of them working, then the binomial theorem applies. It sums all the combinations of some working and some dead subsystems, where the total of working units, r, meets our success criteria,

$$A_{r\ out\ of\ k\ systems} = \sum_{x=r}^{k} \left(\frac{k!}{x!(k-x)!} \right) A_s^x \left(1 - A_s\right)^{k-x} \qquad (8)$$

4. NGAO Availability Block Diagram

In this section, we develop the overall availability block diagram for the NGAO system including science instruments. The subsystems from Table 1 are used to determine the interconnections and redundancy. At the top level, NGAO is composed of a series of blocks as shown in Figure 3. The atmospheric profiler is assumed non-essential for NGAO observations. As such, it is not connected to other blocks in the diagram. The other systems are all essential and non-redundant so they are connected in series. Note that the high-level systems each represent a "single point failure" mode. The diagram is for LGS observations with NGAO. As this is the main science mode, we do not consider NGS observations at this time. We also do not consider the increase in allocated downtime for the LGS facility if NGS observations were used to provide a backup plan,

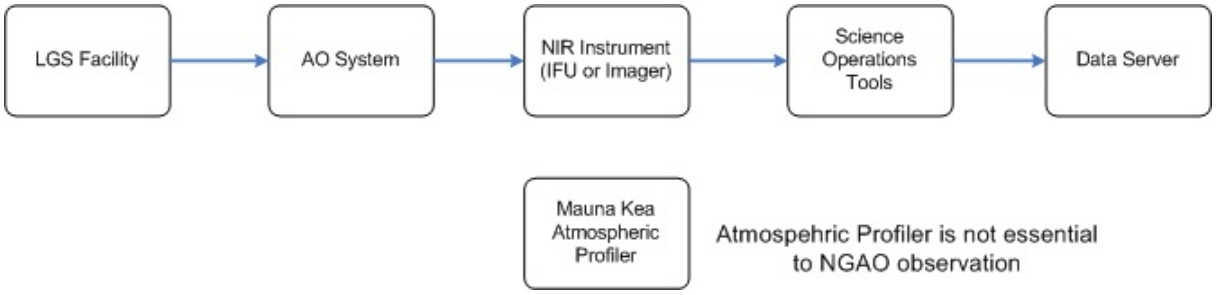

Figure 3: Top-level diagram for NGAO downtime. The Mauna Kea Atmospheric profiler is not essential for NGAO operation so its availability is not included in the uptime calculations.

NGAO can be further decomposed to the next level the product breakdown structure. As shown in Figure 4 the subsystems are connected in series. Two subsystems of the Science Operation Tools are non-essential for the recording of science data. These two subsystems are the Post-Observing Tools and the Pre-Observing Tools. The Post-Observing Tools are only used to reduce data once it is recorded, so a fault in this system only the delays the analysis of the data but does not preclude recording it. The Pre-Observing Tools are used to setup and plan an observation. If these tools are not available is it still possible to proceed with observing although it will no longer be possible to plan and setup new ones. Observations that are already planned or perhaps taken from a backup queue could still proceed. The sub-subsystem availability diagram is shown in Figure 4.

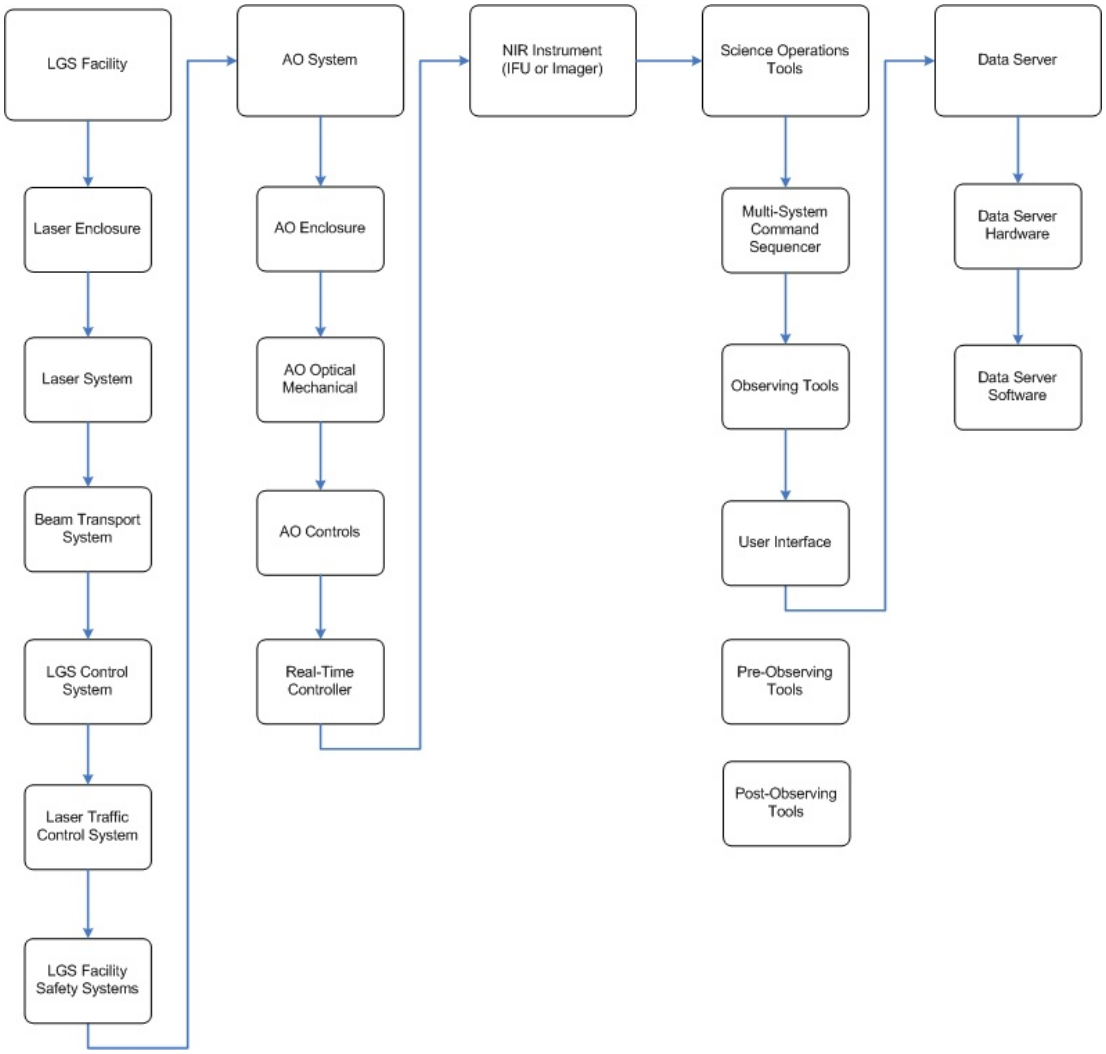

Figure 4: Subsystem level availability diagram for NGAO. Systems that are not essential to record science data are not connected.

5. Overview of Uptime Budget Tool

An inspection of the diagrams in Figure 3 and Figure 4 shows that the main subsystems on NGAO are connected in series and have no redundancy. As such, it is straightforward to analyze the overall system downtime by combining estimates for each subsystem for MTTF and MTTR. We have performed such calculations and they are contained in an Excel [6] spreadsheet. The spreadsheet is organized in a hierarchy with the top-level system downtime sheet that summarizes the aggregated downtime built up from allocations for the availability of individual subsystems of NGAO. The input cells are color-coded and a "stub" sheet is provided to allow increasingly finer grained estimates of availability and downtime to be made as the design of NGAO proceeds.

6. Initial Subsystem Allocations

The subsystem allocations from the availability spreadsheet resulted in the following downtime and availability percentages as shown in Table 5. The instruments are given identical allocations of 1.2% downtime. Only one instrument availability value is used to calculate the overall system availability. This is consistent with the current design decision to have them both in the same Dewar and sharing components to save costs. These are preliminary allocations and are not the result of a detailed calculation of MTTF and MTTR for each subsystem.

NGAO Summary		
PBS Item	**Downtime**	**Availability**
AO System	1.497%	98.5030%
LGS Facility	1.905%	98.0947%
Science Operations Tools	0.230%	99.7696%
Data Server	0.241%	99.7588%
MK Atmospheric Profile (N/A)		
NIR Imager	1.211%	98.7894%
NIR IFS	1.211%	98.7894%
System Totals	4.993%	95.0067%

Table 5: Summary of NGAO downtime allocations. Values were chosen to meet the overall downtime requirement of 5%. The instruments are given identical allocations of 1.2% downtime. Only one instrument is used to calculate the overall system downtime. See main text for explanation.

References

1. "Next Generation Adaptive Optics: System Requirements Document", Version 1.20, Keck Adaptive Optics Note 456, March 27, 2008.
2. Mean time between failures, Wikipedia: http://en.wikipedia.org/wiki/Mean_time_between_failures.
3. MTBF (Mean Time Between Flareups, er, Failures), http://www.faqs.org/faqs/arch-storage/part2/section-151.html.
4. Reliability and Availability Basics, Event Helix: http://www.eventhelix.com/RealtimeMantra/FaultHandling/reliability_availability_basics.htm.
5. Redundancy in engineering, Wikipedia: http://en.wikipedia.org/wiki/Redundancy_(engineering).
6. Uptime calculation spreadsheet, UptimeTool_v1.0.xls.