Dependability Cases

Charles B. Weinstock
John B. Goodenough
John J. Hudak

May 2004

Performance-Critical Systems

Technical Note
CMU/SEI-2004-TN-016

This work is sponsored by the U.S. Department of Defense.

The Software Engineering Institute is a federally funded research and development center sponsored by the U.S. Department of Defense. The authors also wish to acknowledge support from the NASA High Dependability Computing Program under cooperative agreement NCC-2-1298.

Contents

List of Figures

Abstract

Many large software systems display fragility or a lack of dependability caused by inattention to details at various stages of development (e.g., missing data, undocumented assumptions, lack of testing), resulting in a failure to catch errors. This technical note explains how to create a dependability case for a system that helps identify and keep track of such details. A dependability case is defined here as a structured argument providing evidence that a system meets its specified dependability requirements. The technical note describes how to structure the argument and present evidence to support it. A sample problem is presented, as well as issues raised by that problem and future goals.

1 Introduction

In 1999 the President's (Clinton) Information Technology Advisory Committee issued a report that included the following statement:

> *Software is the new physical infrastructure of the information age. It is fundamental to economic success, scientific and technical research, and national security. The Nation needs robust systems, but the software our systems depend on is fragile. Software fragility is its tendency not to work properly—or at all—for long enough periods of time or in the presence of uncontrollable environmental variation. Fragility is manifested as unreliability, lack of security, performance lapses, errors, and difficulty in upgrading. … We have become dangerously dependent on large software systems whose behavior is not well understood and which often fail in unpredicted ways* [PITAC 99].

In 1996 the maiden flight of the European Space Agency's Ariane-5 heavy-lift rocket ended in failure. This failure occurred in spite of the effort that went into making the system dependable. The hardware was redundant and the relevant software, certified as trustworthy during the successful development of the Ariane-4, was reused unchanged. Indeed, it was not considered wise to change software that had worked well. However, Ariane-5 had a significantly different flight envelope than did Ariane-4 and an unhandled software exception caused the rocket to self-destruct. This exception resulted from an overflow that occurred during the conversion of a 64-bit floating-point number to a 16-bit signed integer value.

The error was missed at several stages of development. It was not caught in unit testing because no trajectory data was provided in the requirements. The error was not caught in integration testing because such testing was considered to be difficult and expensive, and the software was considered reliable. The error was not caught by inspection because the implementation assumptions were not documented.

This is but one of many examples of software problems that could have been prevented had sufficient attention been paid to the details. However, there are lots of "details" in a large system, and it is not always obvious which ones are important to the dependable operation of the system. Furthermore, it is difficult to keep track of all of the details even if you can identify them. These problems, and other related problems, can be dealt with by creating a *dependability case* for the system.

2 The Dependability Case

Before we can define a dependability case, we have to define what it means for a system to be dependable. We take the broad view of dependability as defined by the International Federation for Information Processing Working Group 10.4 on Dependable Computing and Fault Tolerance [Laprie 92].

> *Dependability is defined as the trustworthiness of a computer system such that reliance can justifiably be placed on the service it delivers. The service delivered by a system is its behavior as it is perceived by its user(s); a user is another system (human or physical) which interacts with the former.*

There are many attributes of dependability including availability, reliability, safety, security, real-time performance, interoperability, and others. Not all of these attributes will be important to the same degree in every fielded system. For instance, in one system safety may be paramount, even at the expense of availability. In another, security concerns may outweigh performance.

With this in mind, we define a dependability case to be

> a structured argument providing evidence that a system meets its specified dependability requirements.

The dependability requirements of a system include the dependability attributes of interest in the particular system (e.g., security, real-time performance), and the anticipated usage (how and where) of the system. An argument for a system that's being used in a computer on a desktop in an office probably won't suffice if that same system is embedded in a spacecraft in transit to Mars.

The key to a dependability case is the structure of the argument and the evidence that supports the argument. The dependability case can be formal or informal depending upon the requirements, but it must be able to convince a skeptical reviewer of its validity. It becomes a key element in the documentation of the system.

This is similar in spirit to the definition of a safety case as presented by Adelard [Adelard 03]. The dependability case broadens the concept of a safety case to the whole milieu of dependability. Notice that if the only dependability attribute of interest is safety, then a dependability case becomes a safety case.

2.1 Evidence

The evidence surrounding an argument and the ability to reason from it are key to making a credible dependability case. Without evidence of a dependability claim's correctness, there is no way to substantiate the claim. Unfortunately, evidence comes in many different forms, so it is impossible to dictate what kind of evidence or argument is appropriate for every situation.

Some claims can be taken as given in most cases. Consider, for instance, a claim that the speed of light in a vacuum is 299,792,458 meters per second. There is hard evidence to show that this value is correct, and most would trust a claim that depended on this number without additional evidence. Of course it would be a different story if the light were not traveling through a vacuum.

More typically, justifying a claim will require the development of evidence and inferential reasoning built upon that evidence. The evidence may be more or less formal, as long as it is convincing.

Whole books have been written about evidence and how it is used. In his book on evidential foundations for reasoning, for example, David Schum takes a probabilistic view of evidence [Schum 94]. Evidence in support of a claim will tend to increase our estimate of the probability that a claim is true. The better the evidence and the chain of inferential reasoning that utilizes the evidence, the more likely it is that we'll believe a claim to be true. This may seem obvious, but it's the key to determining whether or not a dependability case is convincing.

2.2 How Dependability Cases are Developed

A dependability case is made by making claims about a system and then showing evidence that those claims are valid. Here is an example of a claim:

> "The system is dependable."

This statement is not useful by itself; we need to know which dependability properties are important in this particular system. A more appropriate claim would be

> "The system meets its Dependability Requirements as detailed in document XXX."

This is much better because it tells us (if we refer to document XXX) what it means for the system to be dependable.

Given a claim, we need to articulate a strategy to use to prove that the claim is true. In this case a strategy might be

> "Show that each of the dependability requirements is met individually. Then show that they are met collectively."

This leads to sub-claims, at least one for each of the dependability requirements, with a strategy and sub-sub-claims for each. Eventually the argument gets down to ground truth. This ground truth might be a formal proof, a law of physics, or perhaps even an exhaustive enumeration of possibilities. Once every sub-claim is successfully driven down to its solution we have an argument that the original claim has been satisfied. This argument can be referred to whenever a question about the claim is raised. In particular, it can be used to identify potential problems when a change in the system is contemplated.

2.3 The Problem with Textual Dependability Cases

As should be readily apparent from the above, constructing a dependability case for any real system is going to require a significant number of claims, strategies, and solutions. Organizing the case so that it can easily be reviewed can present major difficulties. The solution to this problem is to notice that the development of the case implies a structure. Claims require proof strategies that lead to sub-claims that eventually lead to evidence supporting the claim. A notation has been developed that takes advantage of this implied structure. Goal Structuring Notation or GSN was developed by Tim Kelly as a means of documenting safety cases [Kelly 98]. We adopt and adapt that notation (and his Visio-based tools) for dependability cases as well. We'll describe the notation in some detail in a following section.

3 Sample Problem

Perhaps the easiest way to describe Goal Structuring Notation is through a detailed example. For this purpose we have chosen the problem of synchronizing the clock in a low-orbiting satellite with the clock in its ground control station. The solution is the User Spacecraft Clock Calibration System (USCCS) as developed by NASA Goddard [Goddard 99].

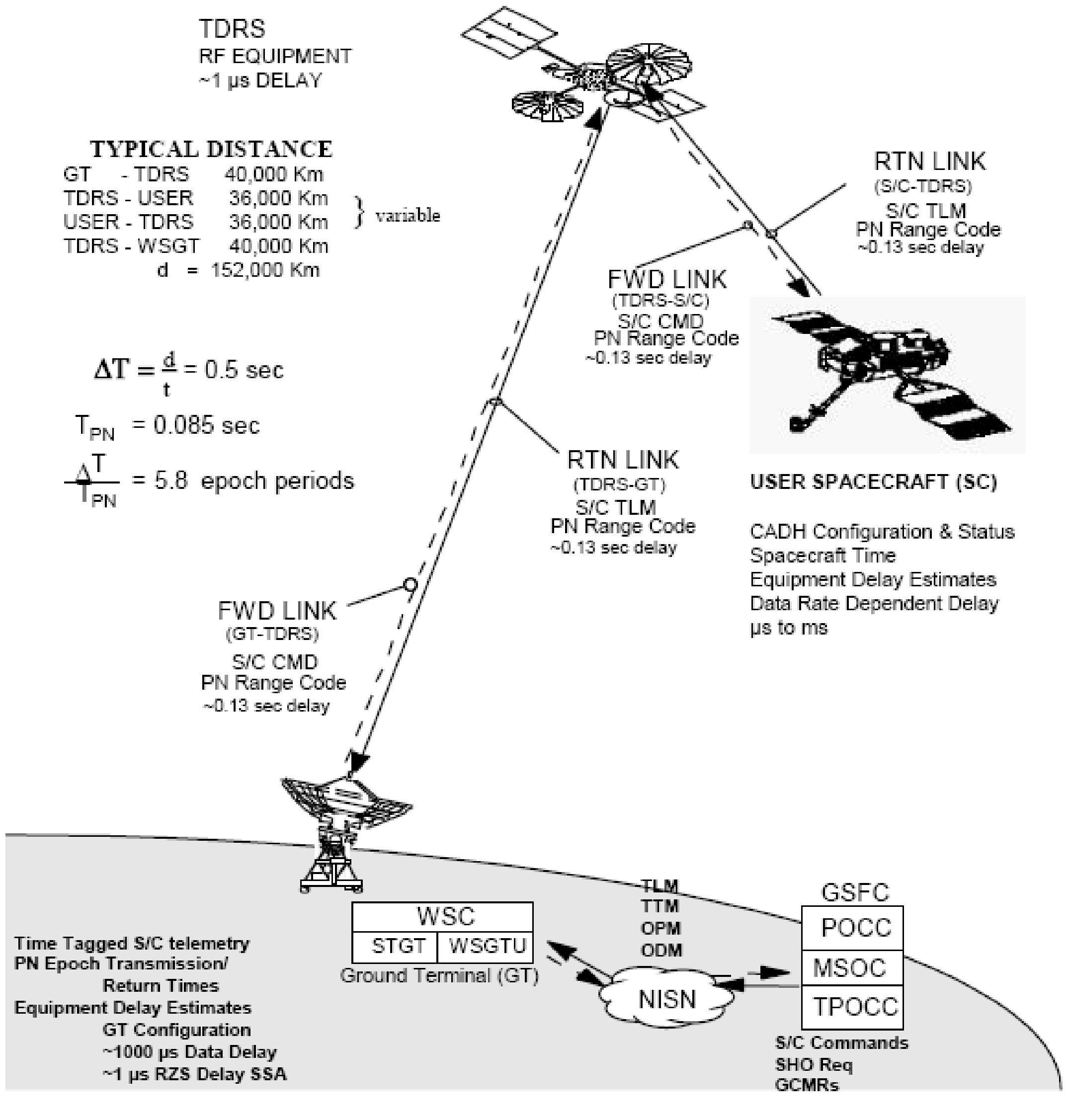

TDRS
RF EQUIPMENT
~1 µs DELAY
TYPICAL DISTANCE
GT - TDRS 40,000 Km
TDRS - USER 36,000 Km
USER - TDRS 36,000 Km
TDRS - WSGT 40,000 Km
d = 152,000 Km
variable
ΔT = d/t = 0.5 sec
T_PN = 0.085 sec
ΔT/T_PN = 5.8 epoch periods
RTN LINK (S/C-TDRS) S/C TLM PN Range Code ~0.13 sec delay
FWD LINK (TDRS-S/C) S/C CMD PN Range Code ~0.13 sec delay
RTN LINK (TDRS-GT) S/C TLM PN Range Code ~0.13 sec delay
FWD LINK (GT-TDRS) S/C CMD PN Range Code ~0.13 sec delay
USER SPACECRAFT (SC)
CADH Configuration & Status
Spacecraft Time
Equipment Delay Estimates
Data Rate Dependent Delay
µs to ms
Time Tagged S/C telemetry
PN Epoch Transmission/ Return Times
Equipment Delay Estimates
GT Configuration
~1000 µs Data Delay
~1 µs RZS Delay SSA
WSC
STGT
WSGTU
Ground Terminal (GT)
TLM
TTM
OPM
ODM
NISN
GSFC
POCC
MSOC
TPOCC
S/C Commands
SHO Req
GCMRs

Figure 1: System Requirements and Overview
(from Goddard 99)

Figure 1, taken from that document, shows the problem that the example is trying to solve. A ground station clock at White Sands Center (WSC) is synchronized with a standard clock. Clock readings are transmitted to and from the User Space Craft (SC) (the low-earth orbiting satellite) through a relay satellite (TDRS), from/to WSC. WSC needs to determine to what extent the SC clock varies from atomic time as known to WSC.

The solution uses the fact that radio waves travel at the speed of light and depends on knowing the minimum and maximum distances that the signal travels. It also depends upon being able to measure delays in signal propagation caused by the equipment (at WSC, TDRS, and the SC) that the signal must traverse. Figure 2 shows a portion of a dependability case for this sample problem.

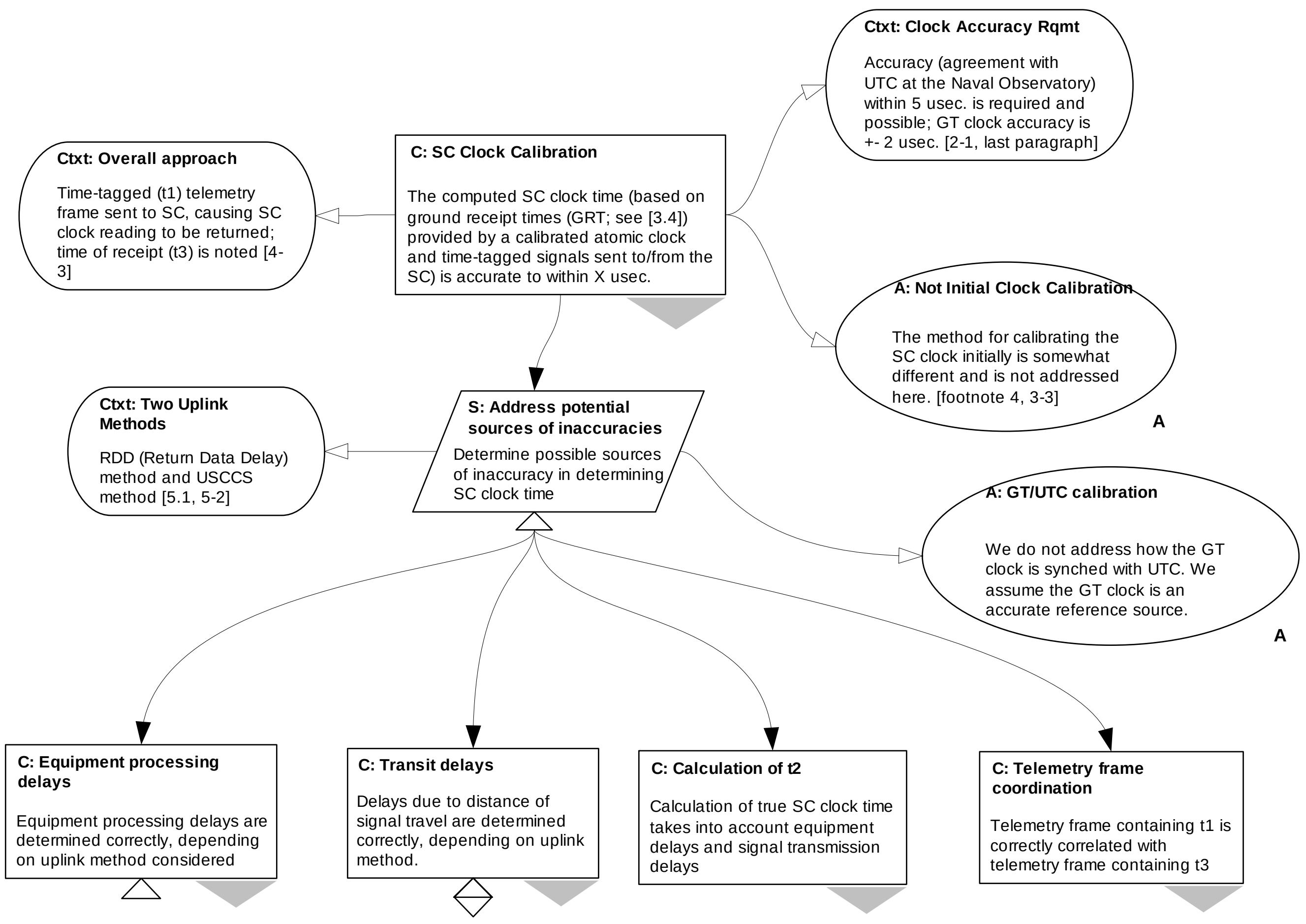

Ctxt: Clock Accuracy Rqmt
Accuracy (agreement with UTC at the Naval Observatory) within 5 usec. is required and possible; GT clock accuracy is +- 2 usec. [2-1, last paragraph]
Ctxt: Overall approach
Time-tagged (t1) telemetry frame sent to SC, causing SC clock reading to be returned; time of receipt (t3) is noted [4-3]
C: SC Clock Calibration
The computed SC clock time (based on ground receipt times (GRT; see [3.4]) provided by a calibrated atomic clock and time-tagged signals sent to/from the SC) is accurate to within X usec.
A: Not Initial Clock Calibration
The method for calibrating the SC clock initially is somewhat different and is not addressed here. [footnote 4, 3-3]
A
Ctxt: Two Uplink Methods
RDD (Return Data Delay) method and USCCS method [5.1, 5-2]
S: Address potential sources of inaccuracies
Determine possible sources of inaccuracy in determining SC clock time
A: GT/UTC calibration
We do not address how the GT clock is synched with UTC. We assume the GT clock is an accurate reference source.
A
C: Equipment processing delays
Equipment processing delays are determined correctly, depending on uplink method considered
C: Transit delays
Delays due to distance of signal travel are determined correctly, depending on uplink method.
C: Calculation of t2
Calculation of true SC clock time takes into account equipment delays and signal transmission delays
C: Telemetry frame coordination
Telemetry frame containing t1 is correctly correlated with telemetry frame containing t3

Figure 2: A Portion of a Dependability Case

The dependability case begins with a claim about the property we are trying to show. The USCCS algorithms are said to keep the SC clock synchronized to within five microseconds of Universal Coordinated Time as measured on the Naval Observatory's atomic clock. So the initial claim is just that: The computed spacecraft clock time is accurate to the actual time to within some number, X, microseconds. As shown, claims are represented in GSN as rectangles.

The case also gives some context at this level, presented in rounded rectangles in GSN. The first piece of context (labeled "Ctxt: Overall approach") is meant to give an overview of the time calibration activity. Its purpose is to make it easier to understand the claim that is stated. The second piece of context ("Ctxt: Clock Accuracy Rqmt") is a statement of the required accuracy of the SC clock, and what it means to be accurate. In this case a reference to the requirements document ("2-1, last paragraph") is also provided. References to specific sections, pages, or paragraphs in the USCCS document are sprinkled throughout the dependability case.

There is also an important assumption that must be stated from the outset, namely that this method is not used for initial clock calibration. It is important to specifically state all assumptions when developing a dependability case. Assumptions are represented by an oval annotated with the letter "A" in GSN.

Given the claim, context, and assumptions, the next step in developing a dependability case is to articulate a strategy for "proving" the claim. Strategies are represented by parallelograms. In this case our strategy is to consider ways in which the synchronization can go wrong ("S: Address potential sources of inaccuracies"). For purposes of our argument, we include the context that there are two uplink methods used—any argument will need to address both of the methods. This is indicated by the up-pointing triangle attached to the parallelogram. There is also an assumption: that the GT clock is already synchronized with UTC. Documenting this assumption is important; without it the strategy presented is incomplete. For completion we'd need to make a claim about the GT synchronization and fully develop it. With the assumption documented, an interested party can go off and explore that issue independently if so desired.

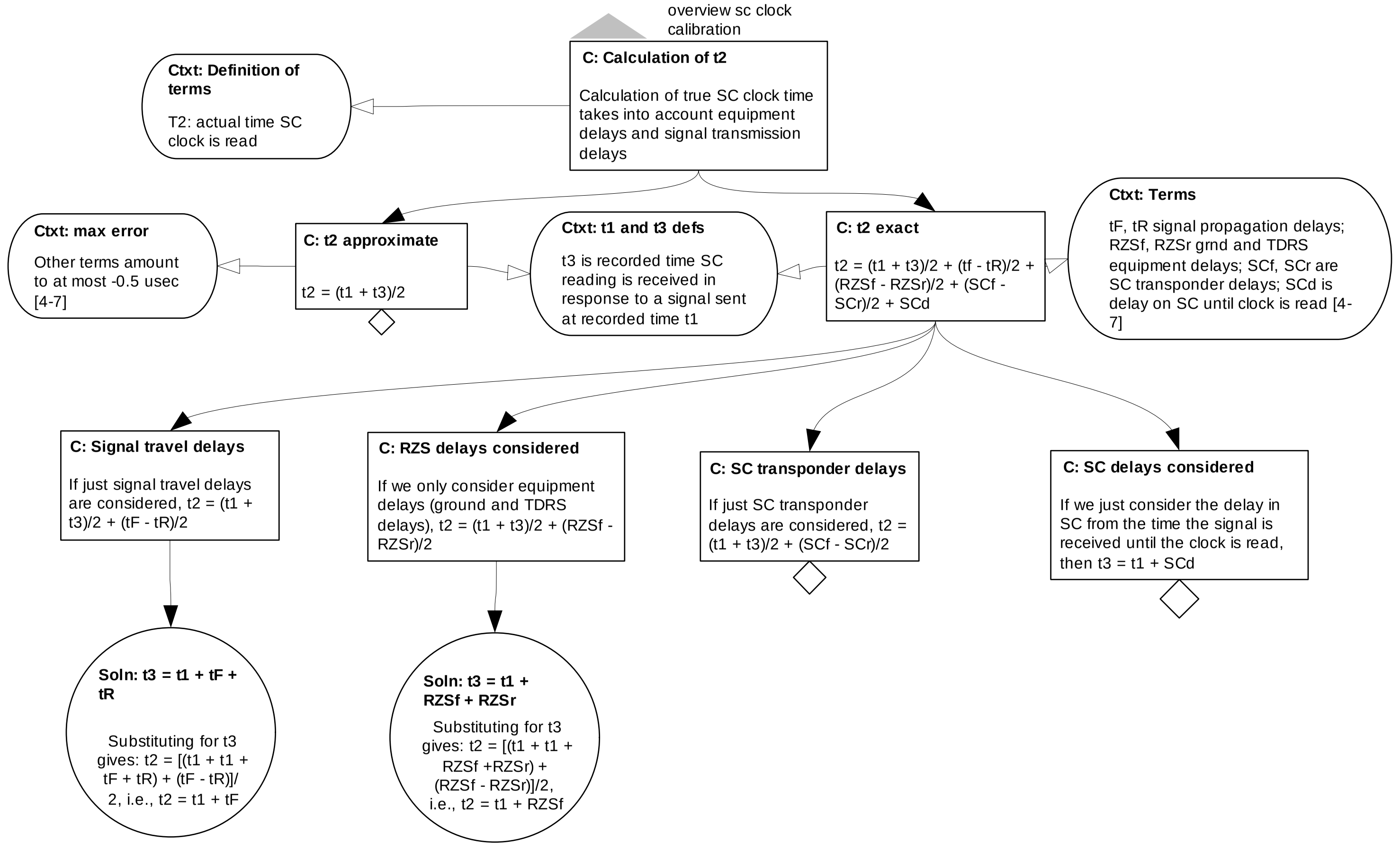
In our example, the identified possible sources of inaccuracy are

- equipment processing delays
- delays due to the distance between the ground, the TDRS, and the SC
- the actual calculation of the true SC clock time
- the ability to coordinate a signal sent to the SC with a signal returned from the SC

Each of these sources of inaccuracy is represented by sub-claims. If we can show that all of the sub-claims are true, then we have shown that our initial claim "C: SC Clock Calibration" is also true.

The complete dependability case has sub-dependability cases for each of the sub-claims. The shaded triangles in Figure 2 indicate that they can be found on following pages. The diamond attached to "C: Transit delays" shows that the proof of the claim is incomplete and further expansion will be necessary. The details of all of the claims can be found in the Appendix. For purposes of exposition we'll follow "C: Calculation of t2" here.

As shown in Figure 3, the context of this claim gives the definition of t2 (the actual time at which SC is read), while other contexts give the definition of t1 (the actual time at which a signal is sent from the ground) and t3 (the actual time at which the signal corresponding to t3 is received on the ground). All of the symbols and boxes have been explained previously with the exception of the circles at the bottom of the dependability case. These leaf-nodes of the dependability case are termed “solutions” because no further expansion is necessary. In this particular example the solutions are algebraic substitutions that “prove” the claims immediately above them. Of course a solution can take on many forms—mathematical proof, exhaustive enumeration, simulation results, and so on.



overview sc clock calibration
C: Calculation of t2
Calculation of true SC clock time takes into account equipment delays and signal transmission delays
Ctxt: Definition of terms
T2: actual time SC clock is read
Ctxt: max error
Other terms amount to at most -0.5 usec [4-7]
C: t2 approximate
t2 = (t1 + t3)/2
Ctxt: t1 and t3 defs
t3 is recorded time SC reading is received in response to a signal sent at recorded time t1
C: t2 exact
t2 = (t1 + t3)/2 + (tf - tR)/2 + (RZSf - RZSr)/2 + (SCf - SCr)/2 + SCd
Ctxt: Terms
tF, tR signal propagation delays; RZSf, RZSr grnd and TDRS equipment delays; SCf, SCr are SC transponder delays; SCd is delay on SC until clock is read [4-7]
C: Signal travel delays
If just signal travel delays are considered, t2 = (t1 + t3)/2 + (tF - tR)/2
C: RZS delays considered
If we only consider equipment delays (ground and TDRS delays), t2 = (t1 + t3)/2 + (RZSf - RZSr)/2
C: SC transponder delays
If just SC transponder delays are considered, t2 = (t1 + t3)/2 + (SCf - SCr)/2
C: SC delays considered
If we just consider the delay in SC from the time the signal is received until the clock is read, then t3 = t1 + SCd
Soln: t3 = t1 + tF + tR
Substituting for t3 gives: t2 = [(t1 + t1 + tF + tR) + (tF - tR)]/2, i.e., t2 = t1 + tF
Soln: t3 = t1 + RZSf + RZSr
Substituting for t3 gives: t2 = [(t1 + t1 + RZSf +RZSr) + (RZSf - RZSr)]/2, i.e., t2 = t1 + RZSf

Figure 3: Expansion of a Sub-Claim

4 Issues Raised by the Example

4.1 Completeness of the Dependability Case

The example illustrates the use of the dependability case methodology, but it leaves some questions unanswered. For instance, the portion of the case shown in Figure 2 asserts that the algorithms used will provide an accurate synchronization of the spacecraft clock with UTC at the ground station. The dependability case asserts that this claim is true because all of the associated sub-claims are true. It implies that there are no other things to be considered in substantiating this claim. How do we know this is true?

An answer, perhaps, can be gleaned from current practices in developing reliable systems. An early step in the development of such systems is often a fault-tree analysis (FTA). The FTA is an exhaustive look at the ways that things can go wrong (not just in the software) and how to mitigate the problems when they do occur. The example dependability case has this flavor. Instead of building the case upon components of the algorithm, it builds the case upon showing how the potential sources of inaccuracy are mitigated. The FTA concept can also be applied to other dependability attributes. When trying to prove real-time performance, for example, it is useful to look at ways in which the schedule might not be met.

4.2 The Bulkiness of the Dependability Case

The USCCS example shown in the appendix runs to 10 pages and it is not complete—there are a significant number of places that need additional expansion to complete the case. Even though it is incomplete, it is relatively structured. (We feel this makes it a more understandable and easier-to-review description of the synchronization method than the Goddard document, which runs nearly 50 pages.) Nevertheless, the bulkiness represents a potential problem.

The Kelly GSN Visio tool helps in this regard by providing an index of claims, strategies, contexts, and so on. The choice of titles for the boxes is important for this reason. With this index and well chosen titles, it is possible for a developer or reviewer to locate the relevant portions of the case reasonably quickly. This makes the dependability case much more useful as a documentation tool.

Note, also, that most dependability cases will refer to a plethora of documents that support the case. Thus, the complete corpus of the dependability case and its supporting evidence is likely to be very large. The only saving grace here is that the documents referred to are needed anyway when designing, implementing, or maintaining the system. The dependability case acts as a roadmap so that this documentation is actually useful for those purposes.

4.3 The Expense of a Dependability Case

As for many verification and validation activities, the development of safety cases can be quite expensive. Since a given dependability case potentially covers many additional attributes, it can end up being even more expensive. This is a serious barrier to the adoption of the technique and may cause many to question its viability.

However, the dependability case is structured as a tree. When traversing a tree one can choose to go depth first or breadth first. We equate breadth to the various attributes being considered, and depth to the details of a case for a particular attribute. Adding attributes to the case makes it broader. Adding depth to a case for a particular attribute should give higher confidence that the attribute has met its requirements. Where safety cases are required it is necessary to go quite deep. However, for other attributes a rather shallow case may be sufficient to achieve enough confidence that the requirements have been met.

4.4 The Potential Benefits of a Dependability Case

The structured approach of stating high-level claims that are decomposed into supporting sub-claims helps to focus attention during development and review on issues of critical importance to dependable operation of a system. The method inherently focuses on system issues, helping to identify the role of software in supporting overall system dependability, since system-wide claims and sub-claim decomposition can readily address hardware/software interactions. In addition, the contextual annotations help direct a reviewer to more detailed information. Thus, the dependability case helps in organizing the vast amount of information that must be considered in drawing conclusions about a system's likely dependability.

A significant benefit of dependability cases is at the leaves of the case. A properly constructed case will show how and why test results are considered to support dependability claims. The chain of reasoning connecting these results (evidence) to high-level claims can help in evaluating the confidence a reviewer should have in particular test results. In particular, the development of a dependability case can help in deciding what kinds of tests are most important for confirming that critical areas of a system have been designed and coded correctly.

5 Next Steps

We are just in the beginning stages of our exploration of the dependability case. We are working under the auspices of the High Dependability Computing Program (HDCP) a joint Carnegie Mellon University and NASA program. The purpose of the program is described below:

- Develop technologies to improve the dependability of NASA mission software.
- Create methods for quantifying dependability, with a major goal being to rigorously evaluate proposed dependability solutions.
- Evaluate important dependability attributes of software on realistic NASA-specific and NASA-relevant testbeds.

The testbed we are working with is the Mission Data System, a next-generation planetary exploration system being developed at NASA's Jet Propulsion Laboratory. Our specific goals include

- further developing the concept of a dependability case
- learning how to determine when a dependability case is complete
- finding ways to reduce the potential bulkiness of the dependability case
- calculating a return-on-investment (ROI) for the dependability case. This will help users decide when it is cost effective to use the dependability case and when it is not.
- begin transitioning the dependability case techniques to engineers developing real systems

MDS personnel will be working directly with the Software Engineering Institute team to achieve these and other goals in the context of their real-world system.

Appendix

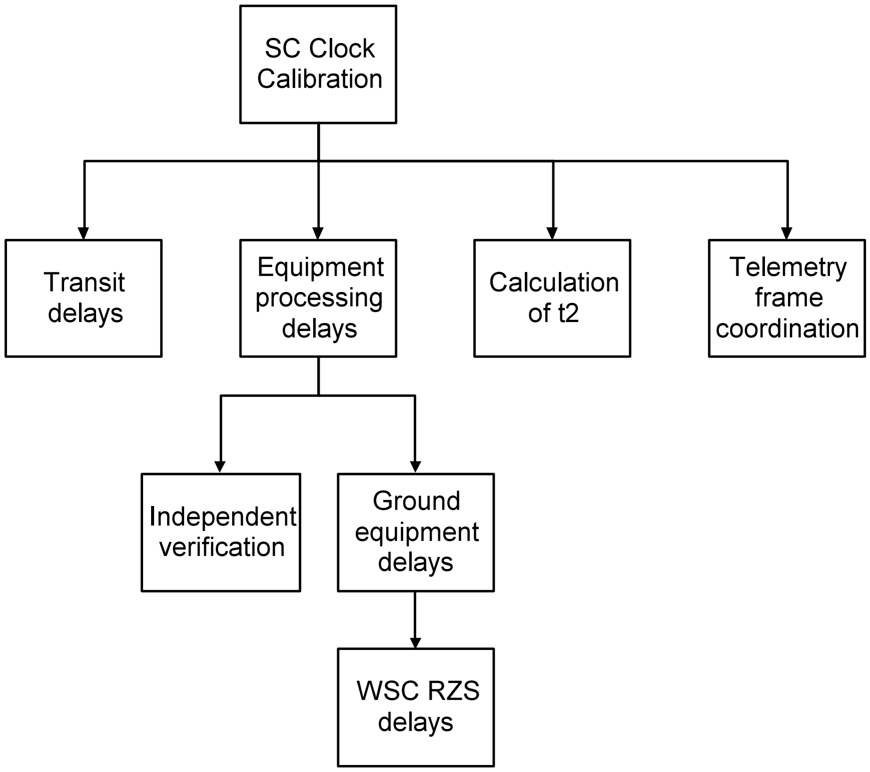


Figure 4: Overview of the USCCS Dependability Case

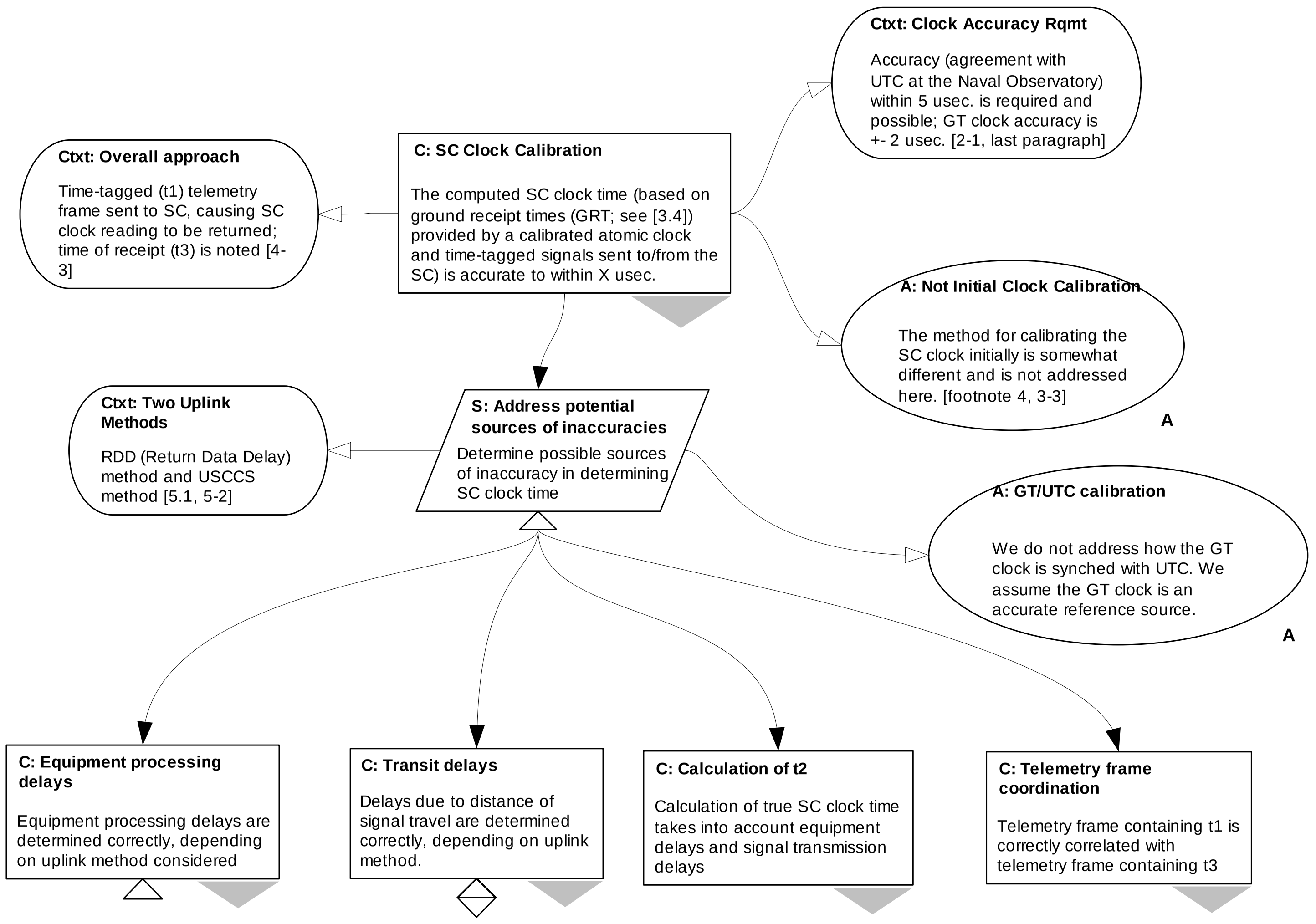

Ctxt: Overall approach
Time-tagged (t1) telemetry frame sent to SC, causing SC clock reading to be returned; time of receipt (t3) is noted [4-3]
C: SC Clock Calibration
The computed SC clock time (based on ground receipt times (GRT; see [3.4]) provided by a calibrated atomic clock and time-tagged signals sent to/from the SC) is accurate to within X usec.
Ctxt: Clock Accuracy Rqmt
Accuracy (agreement with UTC at the Naval Observatory) within 5 usec. is required and possible; GT clock accuracy is +- 2 usec. [2-1, last paragraph]
A: Not Initial Clock Calibration
The method for calibrating the SC clock initially is somewhat different and is not addressed here. [footnote 4, 3-3]
A
Ctxt: Two Uplink Methods
RDD (Return Data Delay) method and USCCS method [5.1, 5-2]
S: Address potential sources of inaccuracies
Determine possible sources of inaccuracy in determining SC clock time
A: GT/UTC calibration
We do not address how the GT clock is synched with UTC. We assume the GT clock is an accurate reference source.
A
C: Equipment processing delays
Equipment processing delays are determined correctly, depending on uplink method considered
C: Transit delays
Delays due to distance of signal travel are determined correctly, depending on uplink method.
C: Calculation of t2
Calculation of true SC clock time takes into account equipment delays and signal transmission delays
C: Telemetry frame coordination
Telemetry frame containing t1 is correctly correlated with telemetry frame containing t3

Figure 5: Top-level USCCS Dependability Case

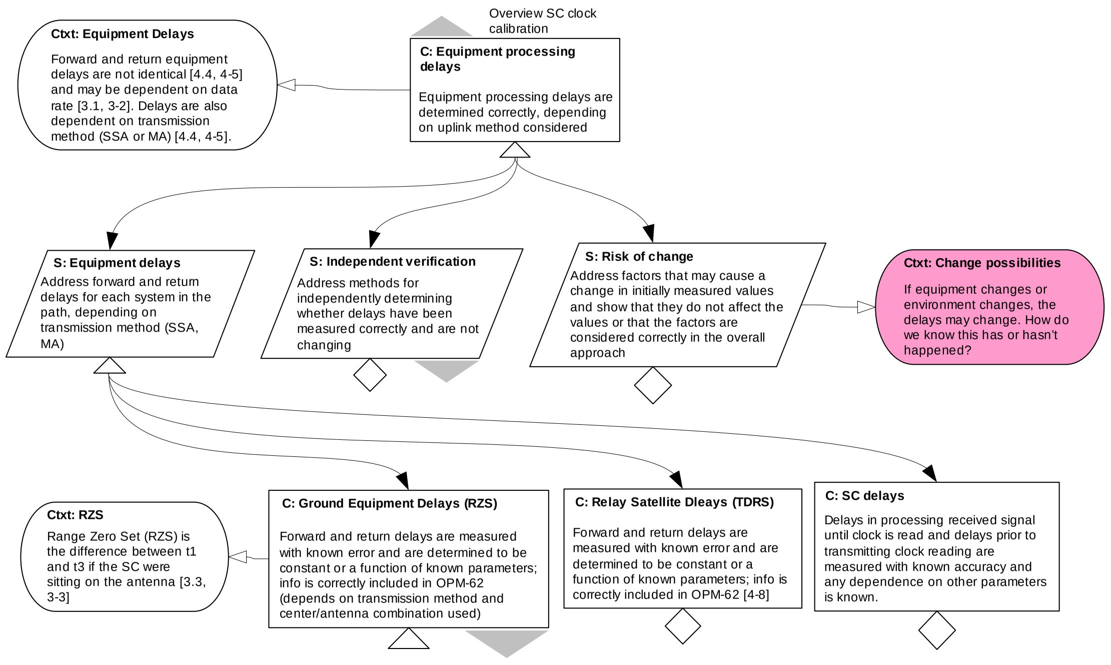

Overview SC clock calibration
C: Equipment processing delays
Equipment processing delays are determined correctly, depending on uplink method considered
Ctxt: Equipment Delays
Forward and return equipment delays are not identical [4.4, 4-5] and may be dependent on data rate [3.1, 3-2]. Delays are also dependent on transmission method (SSA or MA) [4.4, 4-5].
S: Equipment delays
Address forward and return delays for each system in the path, depending on transmission method (SSA, MA)
S: Independent verification
Address methods for independently determining whether delays have been measured correctly and are not changing
S: Risk of change
Address factors that may cause a change in initially measured values and show that they do not affect the values or that the factors are considered correctly in the overall approach
Ctxt: Change possibilities
If equipment changes or environment changes, the delays may change. How do we know this has or hasn't happened?
Ctxt: RZS
Range Zero Set (RZS) is the difference between t1 and t3 if the SC were sitting on the antenna [3.3, 3-3]
C: Ground Equipment Delays (RZS)
Forward and return delays are measured with known error and are determined to be constant or a function of known parameters; info is correctly included in OPM-62 (depends on transmission method and center/antenna combination used)
C: Relay Satellite Dleays (TDRS)
Forward and return delays are measured with known error and are determined to be constant or a function of known parameters; info is correctly included in OPM-62 [4-8]
C: SC delays
Delays in processing received signal until clock is read and delays prior to transmitting clock reading are measured with known accuracy and any dependence on other parameters is known.

Figure 6: Equipment Processing Delays

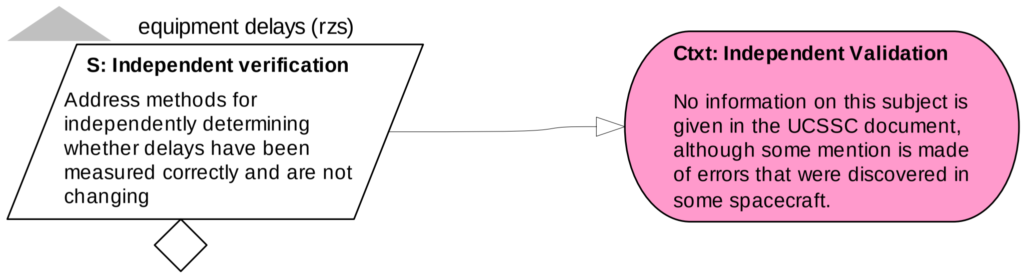
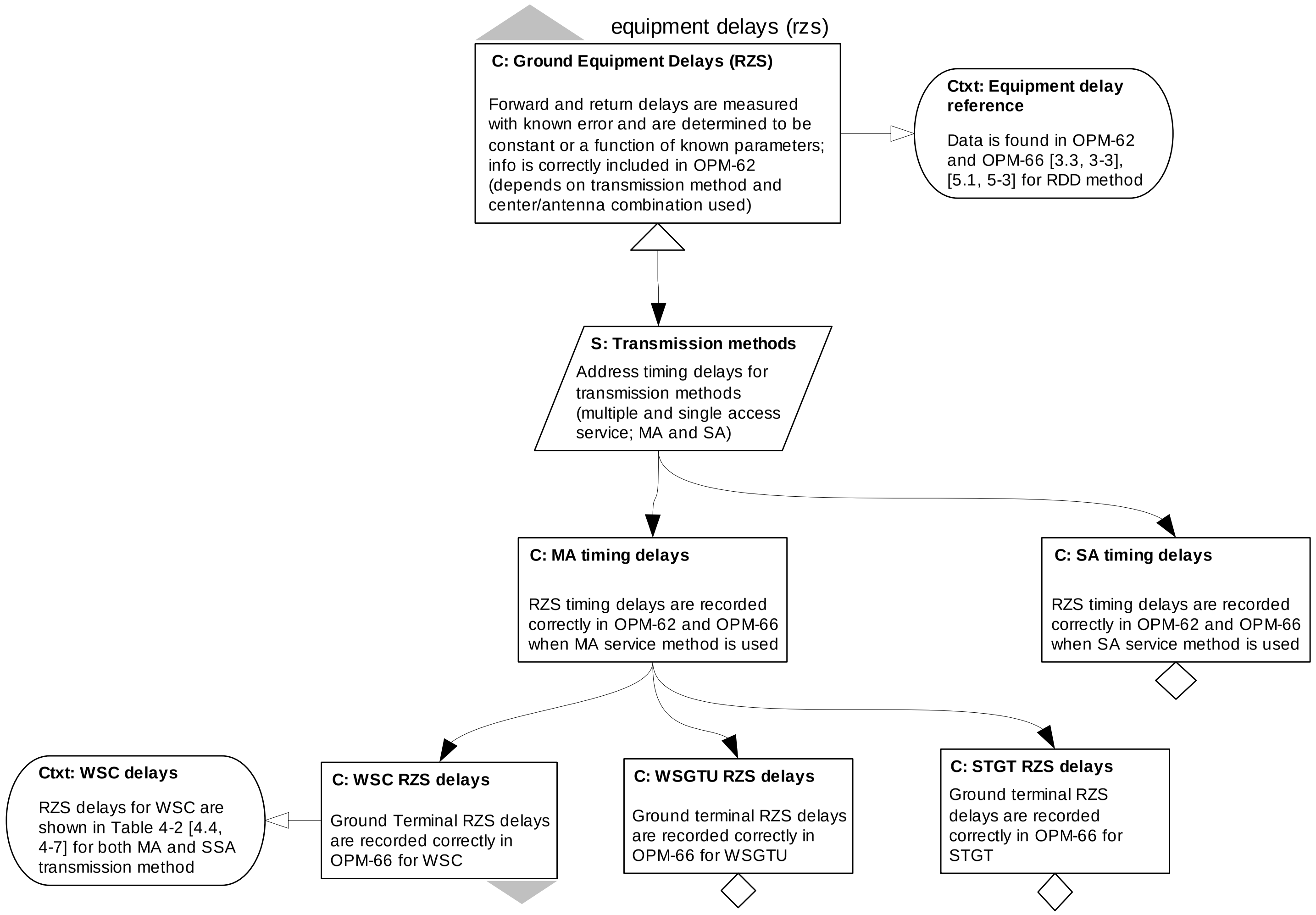

equipment delays (rzs)
S: Independent verification
Address methods for independently determining whether delays have been measured correctly and are not changing
Ctxt: Independent Validation
No information on this subject is given in the UCSSC document, although some mention is made of errors that were discovered in some spacecraft.

Figure 7: Independent Verification of Equipment Delay Measurements



equipment delays (rzs)
C: Ground Equipment Delays (RZS)
Forward and return delays are measured with known error and are determined to be constant or a function of known parameters; info is correctly included in OPM-62 (depends on transmission method and center/antenna combination used)
Ctxt: Equipment delay reference
Data is found in OPM-62 and OPM-66 [3.3, 3-3], [5.1, 5-3] for RDD method
S: Transmission methods
Address timing delays for transmission methods (multiple and single access service; MA and SA)
C: MA timing delays
RZS timing delays are recorded correctly in OPM-62 and OPM-66 when MA service method is used
C: SA timing delays
RZS timing delays are recorded correctly in OPM-62 and OPM-66 when SA service method is used
Ctxt: WSC delays
RZS delays for WSC are shown in Table 4-2 [4.4, 4-7] for both MA and SSA transmission method
C: WSC RZS delays
Ground Terminal RZS delays are recorded correctly in OPM-66 for WSC
C: WSGTU RZS delays
Ground terminal RZS delays are recorded correctly in OPM-66 for WSGTU
C: STGT RZS delays
Ground terminal RZS delays are recorded correctly in OPM-66 for STGT

Figure 8: Ground Equipment Delay (RZS)

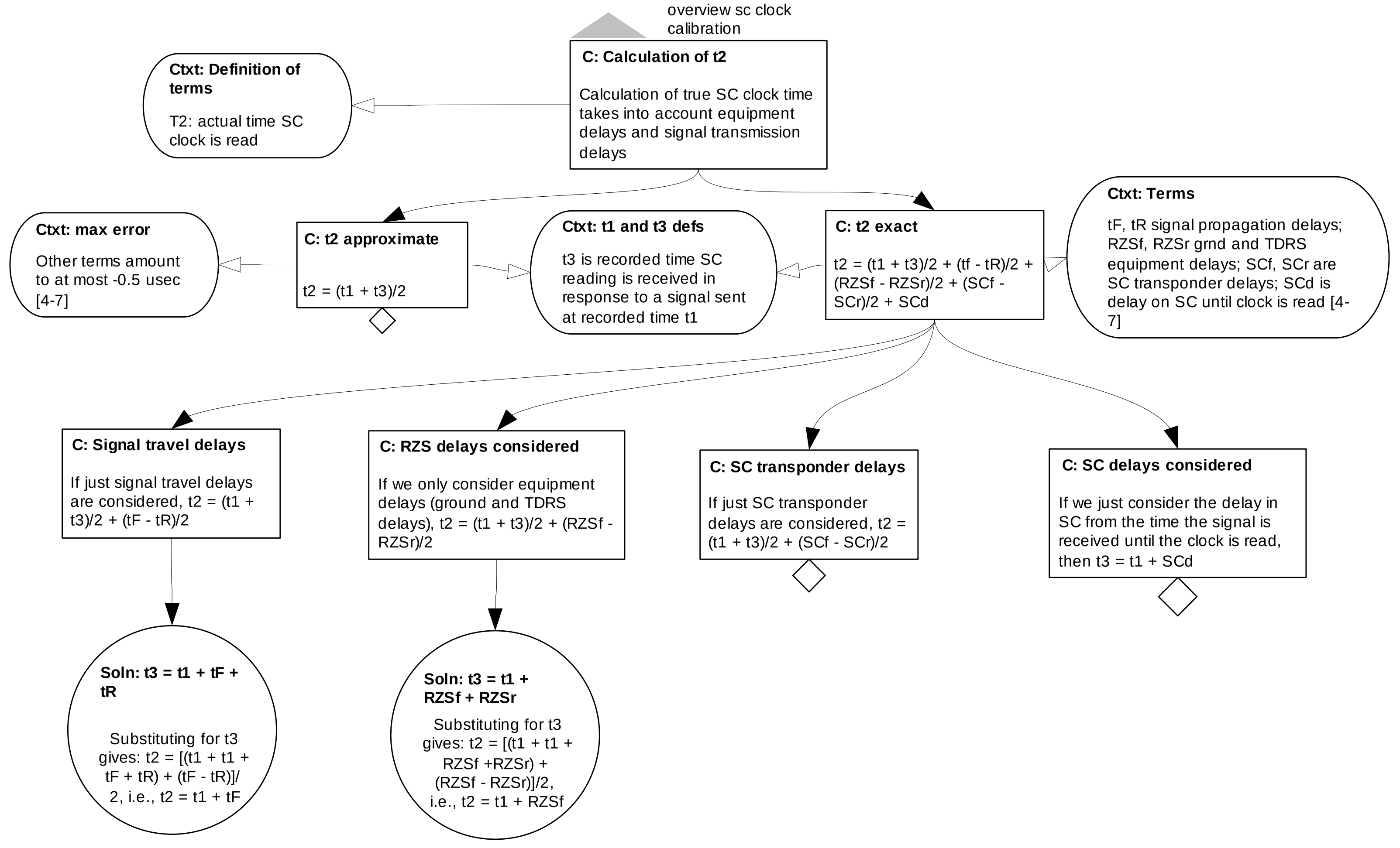

overview sc clock calibration
C: Calculation of t2
Calculation of true SC clock time takes into account equipment delays and signal transmission delays
Ctxt: Definition of terms
T2: actual time SC clock is read
Ctxt: max error
Other terms amount to at most -0.5 usec [4-7]
C: t2 approximate
t2 = (t1 + t3)/2
Ctxt: t1 and t3 defs
t3 is recorded time SC reading is received in response to a signal sent at recorded time t1
C: t2 exact
t2 = (t1 + t3)/2 + (tf - tR)/2 + (RZSf - RZSr)/2 + (SCf - SCr)/2 + SCd
Ctxt: Terms
tF, tR signal propagation delays; RZSf, RZSr grnd and TDRS equipment delays; SCf, SCr are SC transponder delays; SCd is delay on SC until clock is read [4-7]
C: Signal travel delays
If just signal travel delays are considered, t2 = (t1 + t3)/2 + (tF - tR)/2
C: RZS delays considered
If we only consider equipment delays (ground and TDRS delays), t2 = (t1 + t3)/2 + (RZSf - RZSr)/2
C: SC transponder delays
If just SC transponder delays are considered, t2 = (t1 + t3)/2 + (SCf - SCr)/2
C: SC delays considered
If we just consider the delay in SC from the time the signal is received until the clock is read, then t3 = t1 + SCd
Soln: t3 = t1 + tF + tR
Substituting for t3 gives: t2 = [(t1 + t1 + tF + tR) + (tF - tR)]/2, i.e., t2 = t1 + tF
Soln: t3 = t1 + RZSf + RZSr
Substituting for t3 gives: t2 = [(t1 + t1 + RZSf +RZSr) + (RZSf - RZSr)]/2, i.e., t2 = t1 + RZSf

Figure 9: Calculation of t2

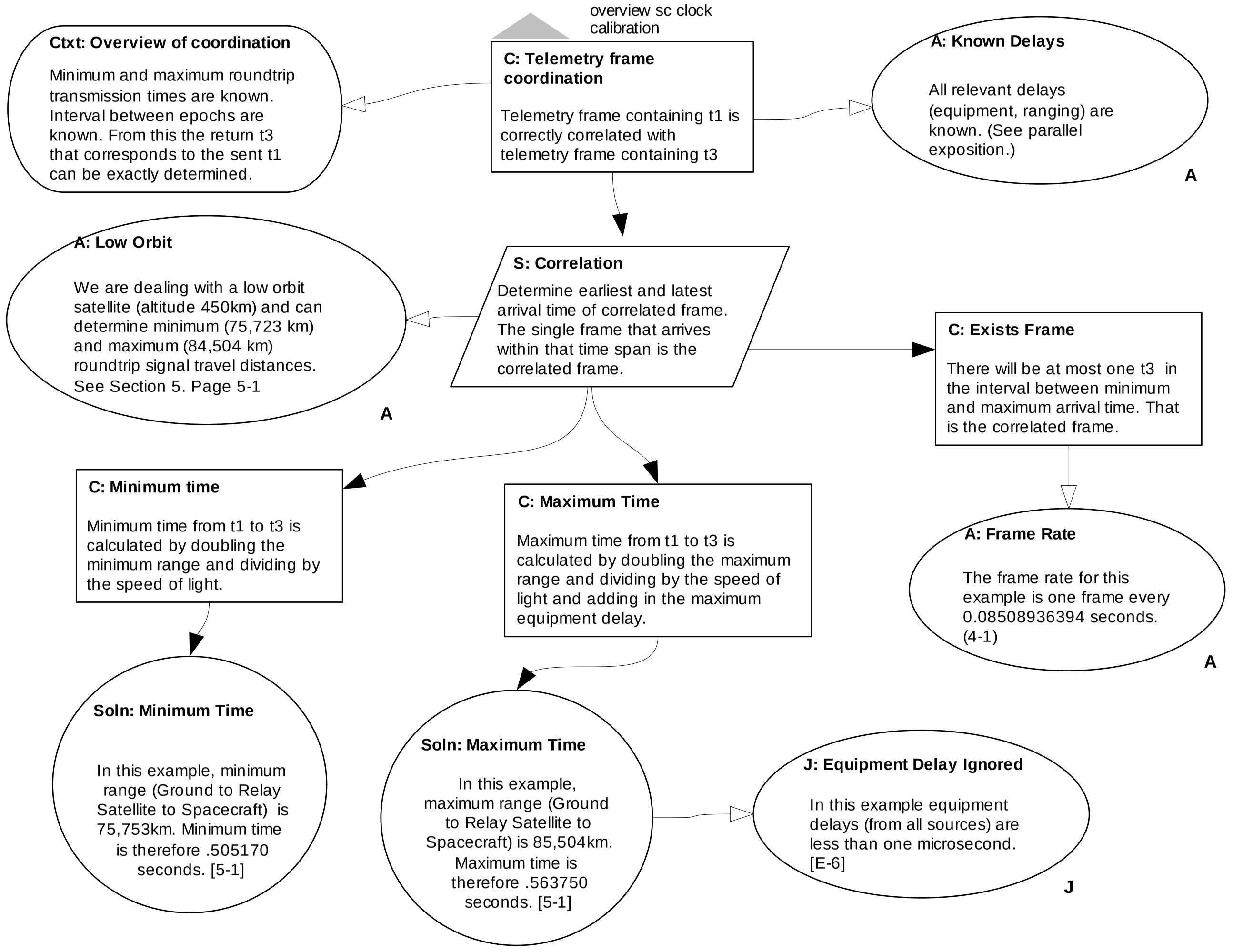

overview sc clock calibration
Ctxt: Overview of coordination
Minimum and maximum roundtrip transmission times are known. Interval between epochs are known. From this the return t3 that corresponds to the sent t1 can be exactly determined.
C: Telemetry frame coordination
Telemetry frame containing t1 is correctly correlated with telemetry frame containing t3
A: Known Delays
All relevant delays (equipment, ranging) are known. (See parallel exposition.)
A
A: Low Orbit
We are dealing with a low orbit satellite (altitude 450km) and can determine minimum (75,723 km) and maximum (84,504 km) roundtrip signal travel distances. See Section 5. Page 5-1
A
S: Correlation
Determine earliest and latest arrival time of correlated frame. The single frame that arrives within that time span is the correlated frame.
C: Exists Frame
There will be at most one t3 in the interval between minimum and maximum arrival time. That is the correlated frame.
C: Minimum time
Minimum time from t1 to t3 is calculated by doubling the minimum range and dividing by the speed of light.
C: Maximum Time
Maximum time from t1 to t3 is calculated by doubling the maximum range and dividing by the speed of light and adding in the maximum equipment delay.
A: Frame Rate
The frame rate for this example is one frame every 0.08508936394 seconds. (4-1)
A
Soln: Minimum Time
In this example, minimum range (Ground to Relay Satellite to Spacecraft) is 75,753km. Minimum time is therefore .505170 seconds. [5-1]
Soln: Maximum Time
In this example, maximum range (Ground to Relay Satellite to Spacecraft) is 85,504km. Maximum time is therefore .563750 seconds. [5-1]
J: Equipment Delay Ignored
In this example equipment delays (from all sources) are less than one microsecond. [E-6]
J

Figure 10: Telemetry Frame Coordination

References

URLs are valid as of the publication date of this document.

[Adelard 03]	*The Adelard Safety Case Development Manual – ASCAD*. London, U.K.: Adelard LLP, 2003.
[Goddard 99]	*User Spacecraft Clock Calibration System (USCCS) Users' Guide.* NASA Goddard, 531-TR-001. Greenbelt, MD: NASA Goddard, 1999.
[Kelly 98]	Kelly, T.P. *Arguing Safety*. Ph.D. Thesis. York, U.K.: University of York, 1998.
[Laprie 92]	Laprie, J.C., Editor. *Dependability: Basic Concepts and Terminology*. Vienna, Austria: Springer-Verlag, 1992.
[PITAC 99]	President's Information Technology Advisory Committee. *Information Technology Research: Investing in Our Future.* <http://www.hpcc.gov/pitac/report/pitac_report.pdf> (1999).
[Schum 94]	Schum, D.A., *Evidential Foundations of Probabilistic Reasoning*. Hoboken, NJ: John Wiley & Sons, 1994.

REPORT DOCUMENTATION PAGE			*Form Approved OMB No. 0704-0188*

1. AGENCY USE ONLY (Leave Blank)	2. REPORT DATE May 2004	3. REPORT TYPE AND DATES COVERED Final
4. TITLE AND SUBTITLE Dependability Cases		5. FUNDING NUMBERS F19628-00-C-0003
6. AUTHOR(S) Charles B. Weinstock, John B. Goodenough, John J. Hudak		
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Software Engineering Institute Carnegie Mellon University Pittsburgh, PA 15213		8. PERFORMING ORGANIZATION REPORT NUMBER CMU/SEI-2004-TN-016
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) HQ ESC/XPK 5 Eglin Street Hanscom AFB, MA 01731-2116		10. SPONSORING/MONITORING AGENCY REPORT NUMBER
11. SUPPLEMENTARY NOTES		
12A DISTRIBUTION/AVAILABILITY STATEMENT Unclassified/Unlimited, DTIC, NTIS		12B DISTRIBUTION CODE

13. ABSTRACT (MAXIMUM 200 WORDS)

Many large software systems display fragility or a lack of dependability caused by inattention to details at various stages of development (e.g., missing data, undocumented assumptions, lack of testing), resulting in a failure to catch errors. This technical note explains how to create a dependability case for a system that helps identify and keep track of such details. A dependability case is defined here as a structured argument providing evidence that a system meets its specified dependability requirements. The technical note describes how to structure the argument and present evidence to support it. A sample problem is presented, as well as issues raised by that problem and future goals.

14. SUBJECT TERMS dependability case, Goal Structuring Notation, GNC			15. NUMBER OF PAGES 30
16. PRICE CODE			
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT UL