Yang Li
COMP 343/443- Fall 2017
Dr. Peter L Dordal
December 8, 2017

WUMP Programming Project
C# version of the project:

Vanilla, Winsize=2

```
Last login: Fri Dec  8 17:00:06 on ttys001
Yangs-MacBook-Air:~ yangli$ clear; cd "/Users/yangli/Desktop/COMP443/YangLiWUMPProject/YangLiwumpt/bin/Debug";
"/Library/Frameworks/Mono.framework/Versions/5.2.0/bin/mono32"  --debug --debugger-
agent=transport=dt_socket,address=127.0.0.1:60999  "/Users/yangli/Desktop/COMP443/YangLiWUMPProject/YangLiwump
t/bin/Debug/wumpt1.exe" ; echo $? > /var/folders/24/2mdjgs2x76zfx9554t61_bxw0000gn/T/tmp761a389b.tmp; echo; read -p
'Press any key to continue...' -n1; exit
```

```
Looking up address of ulam1.cs.luc.edu... got it: 147.126.2.91
req size = 12, filename=vanilla
```

```
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=2
        DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
that you can trust, that you can depend on, that is worthy of your
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=0
        DATA packet blocknum = 2
vy load, and a
reliable transistor is one that operates for years, possibly under
extreme temperature and radiation.

The common enemy of reliability in these examples are adverse
circumstances and influences that may cause a deterioration of the
physical properties of material.  The accumulation of these influences
is called aging.  Reliability is achieved by dimensioning the
mechanisms properly, taking such adverse conditions into
consideration.  In a railway system the schedule is arranged such that
it l
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=1
        DATA packet blocknum = 3
eaves room for catching up on lost time, and an ample supply of
spare engines is kept on the alert for emergencies.  A bridge is built
stronger than actually needed most of the time -- and a transistor is
equipped with cooling devices and radiation shields.

What does all this have to do with software?  Well, we all have
experienced failures of compter systems; and we all would like them to
be reliable too.  When a computer fails, the first question among its
intimates is usually: is the hardware or the sof
sending ACK[1]
```

```
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=590
        DATA packet blocknum = 2
vy load, and a
reliable transistor is one that operates for years, possibly under
extreme temperature and radiation.

The common enemy of reliability in these examples are adverse
circumstances and influences that may cause a deterioration of the
physical properties of material.  The accumulation of these influences
is called aging.  Reliability is achieved by dimensioning the
mechanisms properly, taking such adverse conditions into
consideration.  In a railway system the schedule is arranged such that
it l
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=311
        DATA packet blocknum = 2
vy load, and a
reliable transistor is one that operates for years, possibly under
extreme temperature and radiation.

The common enemy of reliability in these examples are adverse
circumstances and influences that may cause a deterioration of the
physical properties of material.  The accumulation of these influences
is called aging.  Reliability is achieved by dimensioning the
mechanisms properly, taking such adverse conditions into
consideration.  In a railway system the schedule is arranged such that
it l
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=298
        DATA packet blocknum = 2
vy load, and a
reliable transistor is one that operates for years, possibly under
extreme temperature and radiation.

The common enemy of reliability in these examples are adverse
circumstances and influences that may cause a deterioration of the
physical properties of material.  The accumulation of these influences
is called aging.  Reliability is achieved by dimensioning the
mechanisms properly, taking such adverse conditions into
```

```
consideration.  In a railway system the schedule is arranged such that
it l
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=431
        DATA packet blocknum = 2
vy load, and a
reliable transistor is one that operates for years, possibly under
extreme temperature and radiation.

The common enemy of reliability in these examples are adverse
circumstances and influences that may cause a deterioration of the
physical properties of material.  The accumulation of these influences
is called aging.  Reliability is achieved by dimensioning the
mechanisms properly, taking such adverse conditions into
consideration.  In a railway system the schedule is arranged such that
it l
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=367
        DATA packet blocknum = 2
vy load, and a
reliable transistor is one that operates for years, possibly under
extreme temperature and radiation.

The common enemy of reliability in these examples are adverse
circumstances and influences that may cause a deterioration of the
physical properties of material.  The accumulation of these influences
is called aging.  Reliability is achieved by dimensioning the
mechanisms properly, taking such adverse conditions into
consideration.  In a railway system the schedule is arranged such that
it l
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=467
        DATA packet blocknum = 2
vy load, and a
reliable transistor is one that operates for years, possibly under
extreme temperature and radiation.

The common enemy of reliability in these examples are adverse
circumstances and influences that may cause a deterioration of the
```

```
physical properties of material.  The accumulation of these influences
is called aging.  Reliability is achieved by dimensioning the
mechanisms properly, taking such adverse conditions into
consideration.  In a railway system the schedule is arranged such that
it l
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=332
        DATA packet blocknum = 2
vy load, and a
reliable transistor is one that operates for years, possibly under
extreme temperature and radiation.

The common enemy of reliability in these examples are adverse
circumstances and influences that may cause a deterioration of the
physical properties of material.  The accumulation of these influences
is called aging.  Reliability is achieved by dimensioning the
mechanisms properly, taking such adverse conditions into
consideration.  In a railway system the schedule is arranged such that
it l
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=502
        DATA packet blocknum = 2
vy load, and a
reliable transistor is one that operates for years, possibly under
extreme temperature and radiation.

The common enemy of reliability in these examples are adverse
circumstances and influences that may cause a deterioration of the
physical properties of material.  The accumulation of these influences
is called aging.  Reliability is achieved by dimensioning the
mechanisms properly, taking such adverse conditions into
consideration.  In a railway system the schedule is arranged such that
it l
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=313
        DATA packet blocknum = 2
vy load, and a
reliable transistor is one that operates for years, possibly under
extreme temperature and radiation.
```

```
The common enemy of reliability in these examples are adverse
circumstances and influences that may cause a deterioration of the
physical properties of material.  The accumulation of these influences
is called aging.  Reliability is achieved by dimensioning the
mechanisms properly, taking such adverse conditions into
consideration.  In a railway system the schedule is arranged such that
it l
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=383
	DATA packet blocknum = 2
vy load, and a
reliable transistor is one that operates for years, possibly under
extreme temperature and radiation.

The common enemy of reliability in these examples are adverse
circumstances and influences that may cause a deterioration of the
physical properties of material.  The accumulation of these influences
is called aging.  Reliability is achieved by dimensioning the
mechanisms properly, taking such adverse conditions into
consideration.  In a railway system the schedule is arranged such that
it l
sending ACK[1]
hard timeout
hard timeout
```

Vanilla, Winsize=2

Expected_block++

```
Last login: Fri Dec  8 17:00:27 on ttys000
Yangs-MacBook-Air:~ yangli$ clear; cd "/Users/yangli/Desktop/COMP443/YangLiWUMPProject/YangLiwumpt/bin/Debug"; "/Library/Frameworks/Mono.framework/Versions/5.2.0/bin/mono32"  --debug --debugger-agent=transport=dt_socket,address=127.0.0.1:61313  "/Users/yangli/Desktop/COMP443/YangLiWUMPProject/YangLiwumpt/bin/Debug/wumpt1.exe" ; echo $? > /var/folders/24/2mdjgs2x76zfx9554t61_bxw0000gn/T/tmp1fc0901e.tmp; echo; read -p 'Press any key to continue...' -n1; exit
```

```
Looking up address of ulam1.cs.luc.edu... got it: 147.126.2.91
req size = 12, filename=vanilla
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=3
        DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
that you can trust, that you can depend on, that is worthy of your
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[2]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=0
        DATA packet blocknum = 2
vy load, and a
reliable transistor is one that operates for years, possibly under
extreme temperature and radiation.

The common enemy of reliability in these examples are adverse
circumstances and influences that may cause a deterioration of the
physical properties of material.  The accumulation of these influences
is called aging.  Reliability is achieved by dimensioning the
```

mechanisms properly, taking such adverse conditions into consideration. In a railway system the schedule is arranged such that it l

```
sending ACK[3]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=1
        DATA packet blocknum = 3
```

eaves room for catching up on lost time, and an ample supply of spare engines is kept on the alert for emergencies. A bridge is built stronger than actually needed most of the time -- and a transistor is equipped with cooling devices and radiation shields.

What does all this have to do with software? Well, we all have experienced failures of compter systems; and we all would like them to be reliable too. When a computer fails, the first question among its intimates is usually: is the hardware or the sof

```
sending ACK[4]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=0
        DATA packet blocknum = 4
```

tware the culprit?

Most customers of a computation center show signs of relief when the latter is announced, for the disruption of service is then quickly ended by a so-called deadstart, and life goes on as if (almost) nothing had occurred. Indeed there had been neither an earthquake, nor a snowstorm, nor a weighty load, nor heat or radiation. Instead, merely unpredictable circumstances had led to a state of computation for which the logical structure of the program had not been designed, which the system

```
sending ACK[5]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=1
        DATA packet blocknum = 5
```

's designers didn't anticipate. And when pressing the deadstart button, the computer operator is reasonably confident that these circumstances won't reoccur too soon.

What must we conclude? We understand by the term software the collection of programs that deterministically prescribe a system's detailed behaviour and transtions of state. These programs are constants and are independent of any "adverse conditions" of an environment. Hence, software cannot fail because of unpredictable

```
happenings and age,
sending ACK[6]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=0
        DATA packet blocknum = 6
 but only due to defects in its logical design. This
leads us to a replacement of the attribute "reliable" by "correct."

We may be accused of nitpicking with words.  To this I can only reply
that the choice of words often reveals a speaker's attitude more
profoundly than is clear to him.  The attitude through which we content
ourselves at producing "reliable" software instead of correct software,
bears the danger that we may also consider various degrees of
reliability.  Software may then be termed reliabl
sending ACK[7]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=1
        DATA packet blocknum = 7
e and "more reliable";
we may also call it correct, but certainly not "more correct."

The difference in these words is also manifested in the techniques to
be employed in producing reliability in software versus in clocks,
bridges, and transistors.  In most technical phenomena, reliability is
achieved by overdimensioning the components, by using high quality
material, or by supplying standby equipment that automatically goes
into action when a failure occurs.  In programs, merely repeating a
logical test t
sending ACK[8]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=0
        DATA packet blocknum = 8
en times instead of performing it once does not help, if
the logical structure is correct and the underlying hardware is
reliable.  In fact, the degree to which a program is unreliable is
exactly the probability with which its incorrect parts are executed.
But this measure is not a property of the program itself.

Reconciling ourselves with the word correct in place of reliable has
the advantage that we more readily identify the causes of failures of
our products to meet their goal.  They are not to be soug
sending ACK[9]
```

```
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=1
        DATA packet blocknum = 9
ht in
external, unforeseeable, adverse circumstances, but solely in our own
inadequate minds, and in our failure to communicate, if several people
participate in a program's design.  The advantage of this recognition
is that we know where to concentrate our efforts; its unpleasant part
is the fact that it will be a neverending crusade, because committing
mistakes is a truly human characteristic.

The most sensible targets in our drive at producing correct software
are evidently the programmers themselves.
sending ACK[10]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=0
        DATA packet blocknum = 10
Nothing whatsoever can
replace a sound, systematic training in precise reasoning.  Other
sensible targets are the tools that we employ to assist our reasoning.
These include primarily the formal languages in which we express our
thoughts and abstractions, and by which we transmit them to other
people.  We have directed our efforts to improve our programming tools
since more than a decade, and I will therefore devote the main part of
this paper to a report and an evaluation of the latest product, the
languag
sending ACK[11]
rec'd packet: len=66; proto=1; opcode=2; src=(147.126.2.91/4517); time=1
        DATA packet blocknum = 11
e Pascal, in the light of the topic "reliable software."


sending ACK[12]
hard timeout
hard timeout
```

Lose:
Winsize=2

```
Last login: Fri Dec  8 17:05:35 on ttys000
```

```
Yangs-MacBook-Air:~ yangli$ clear; cd "/Users/yangli/Desktop/COMP443/YangLiWUMPProject/YangLiwumpt/bin/Debug";
"/Library/Frameworks/Mono.framework/Versions/5.2.0/bin/mono32"  --debug --debugger-
agent=transport=dt_socket,address=127.0.0.1:62294  "/Users/yangli/Desktop/COMP443/YangLiWUMPProject/YangLiwump
t/bin/Debug/wumpt1.exe" ; echo $? > /var/folders/24/2mdjgs2x76zfx9554t61_bxw0000gn/T/tmp16bb3823.tmp; echo; read -p
'Press any key to continue...' -n1; exit




Looking up address of ulam1.cs.luc.edu... got it: 147.126.2.91
req size = 9, filename=lose
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=5
	DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
that you can trust, that you can depend on, that is worthy of your
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[1]
hard timeout
hard timeout
```

Lose:
Winsize=2
expected_block++

```
Last login: Fri Dec  8 17:01:54 on ttys001
Yangs-MacBook-Air:~ yangli$ clear; cd "/Users/yangli/Desktop/COMP443/YangLiWUMPProject/YangLiwumpt/bin/Debug";
"/Library/Frameworks/Mono.framework/Versions/5.2.0/bin/mono32"  --debug --debugger-
agent=transport=dt_socket,address=127.0.0.1:62113  "/Users/yangli/Desktop/COMP443/YangLiWUMPProject/YangLiwump
t/bin/Debug/wumpt1.exe" ; echo $? > /var/folders/24/2mdjgs2x76zfx9554t61_bxw0000gn/T/tmpdd13d31.tmp; echo; read -p
'Press any key to continue...' -n1; exit
```

```
Looking up address of ulam1.cs.luc.edu... got it: 147.126.2.91
req size = 9, filename=lose
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=4
        DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
```

```
that you can trust, that you can depend on, that is worthy of your
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[2]
hard timeout
hard timeout
hard timeout
```

Spray

```
Last login: Fri Dec  8 17:06:24 on ttys001
Yangs-MacBook-Air:~ yangli$ clear; cd "/Users/yangli/Desktop/COMP443/YangLiWUMPProject/YangLiwumpt/bin/Debug";
"/Library/Frameworks/Mono.framework/Versions/5.2.0/bin/mono32"  --debug --debugger-
agent=transport=dt_socket,address=127.0.0.1:62663  "/Users/yangli/Desktop/COMP443/YangLiWUMPProject/YangLiwump
t/bin/Debug/wumpt1.exe" ; echo $? > /var/folders/24/2mdjgs2x76zfx9554t61_bxw0000gn/T/tmp7292610c.tmp; echo; read -p
'Press any key to continue...' -n1; exit
```

```
Looking up address of ulam1.cs.luc.edu... got it: 147.126.2.91
req size = 10, filename=spray
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=3
        DATA packet blocknum = 1
```

```
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
that you can trust, that you can depend on, that is worthy of your
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=1093
        DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
that you can trust, that you can depend on, that is worthy of your
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=402
        DATA packet blocknum = 2
vy load, and a
reliable transistor is one that operates for years, possibly under
extreme temperature and radiation.

The common enemy of reliability in these examples are adverse
circumstances and influences that may cause a deterioration of the
physical properties of material.  The accumulation of these influences
is called aging.  Reliability is achieved by dimensioning the
mechanisms properly, taking such adverse conditions into
consideration.  In a railway system the schedule is arranged such that
it l
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=401
```

```
        DATA packet blocknum = 2
vy load, and a
reliable transistor is one that operates for years, possibly under
extreme temperature and radiation.

The common enemy of reliability in these examples are adverse
circumstances and influences that may cause a deterioration of the
physical properties of material.  The accumulation of these influences
is called aging.  Reliability is achieved by dimensioning the
mechanisms properly, taking such adverse conditions into
consideration.  In a railway system the schedule is arranged such that
it l
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=401
        DATA packet blocknum = 2
vy load, and a
reliable transistor is one that operates for years, possibly under
extreme temperature and radiation.

The common enemy of reliability in these examples are adverse
circumstances and influences that may cause a deterioration of the
physical properties of material.  The accumulation of these influences
is called aging.  Reliability is achieved by dimensioning the
mechanisms properly, taking such adverse conditions into
consideration.  In a railway system the schedule is arranged such that
it l
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=510
        DATA packet blocknum = 2
vy load, and a
reliable transistor is one that operates for years, possibly under
extreme temperature and radiation.

The common enemy of reliability in these examples are adverse
circumstances and influences that may cause a deterioration of the
physical properties of material.  The accumulation of these influences
is called aging.  Reliability is achieved by dimensioning the
mechanisms properly, taking such adverse conditions into
consideration.  In a railway system the schedule is arranged such that
```

```
it l
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=0
        DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
that you can trust, that you can depend on, that is worthy of your
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=476
        DATA packet blocknum = 2
vy load, and a
reliable transistor is one that operates for years, possibly under
extreme temperature and radiation.

The common enemy of reliability in these examples are adverse
circumstances and influences that may cause a deterioration of the
physical properties of material.  The accumulation of these influences
is called aging.  Reliability is achieved by dimensioning the
mechanisms properly, taking such adverse conditions into
consideration.  In a railway system the schedule is arranged such that
it l
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=416
        DATA packet blocknum = 2
vy load, and a
reliable transistor is one that operates for years, possibly under
extreme temperature and radiation.

The common enemy of reliability in these examples are adverse
circumstances and influences that may cause a deterioration of the
physical properties of material.  The accumulation of these influences
is called aging.  Reliability is achieved by dimensioning the
```

```
mechanisms properly, taking such adverse conditions into
consideration.  In a railway system the schedule is arranged such that
it l
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=308
        DATA packet blocknum = 2
vy load, and a
reliable transistor is one that operates for years, possibly under
extreme temperature and radiation.

The common enemy of reliability in these examples are adverse
circumstances and influences that may cause a deterioration of the
physical properties of material.  The accumulation of these influences
is called aging.  Reliability is achieved by dimensioning the
mechanisms properly, taking such adverse conditions into
consideration.  In a railway system the schedule is arranged such that
it l
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=422
        DATA packet blocknum = 2
vy load, and a
reliable transistor is one that operates for years, possibly under
extreme temperature and radiation.

The common enemy of reliability in these examples are adverse
circumstances and influences that may cause a deterioration of the
physical properties of material.  The accumulation of these influences
is called aging.  Reliability is achieved by dimensioning the
mechanisms properly, taking such adverse conditions into
consideration.  In a railway system the schedule is arranged such that
it l
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=377
        DATA packet blocknum = 2
vy load, and a
reliable transistor is one that operates for years, possibly under
extreme temperature and radiation.

The common enemy of reliability in these examples are adverse
```

```
circumstances and influences that may cause a deterioration of the
physical properties of material.  The accumulation of these influences
is called aging.  Reliability is achieved by dimensioning the
mechanisms properly, taking such adverse conditions into
consideration.  In a railway system the schedule is arranged such that
it l
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=456
        DATA packet blocknum = 2
vy load, and a
reliable transistor is one that operates for years, possibly under
extreme temperature and radiation.

The common enemy of reliability in these examples are adverse
circumstances and influences that may cause a deterioration of the
physical properties of material.  The accumulation of these influences
is called aging.  Reliability is achieved by dimensioning the
mechanisms properly, taking such adverse conditions into
consideration.  In a railway system the schedule is arranged such that
it l
sending ACK[1]
hard timeout
hard timeout
hard timeout
```

Spray

Expected_block++

```
Last login: Fri Dec  8 17:08:04 on ttys000
Yangs-MacBook-Air:~ yangli$ clear; cd "/Users/yangli/Desktop/COMP443/YangLiWUMPProject/YangLiwumpt/bin/Debug"; "/Library/Frameworks/Mono.framework/Versions/5.2.0/bin/mono32"  --debug --debugger-agent=transport=dt_socket,address=127.0.0.1:62907  "/Users/yangli/Desktop/COMP443/YangLiWUMPProject/YangLiwumpt/bin/Debug/wumpt1.exe" ; echo $? > /var/folders/24/2mdjgs2x76zfx9554t61_bxw0000gn/T/tmp215c037f.tmp; echo; read -p 'Press any key to continue...' -n1; exit
```

```
Looking up address of ulam1.cs.luc.edu... got it: 147.126.2.91
req size = 10, filename=spray
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=6
        DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
that you can trust, that you can depend on, that is worthy of your
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[2]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=1612
        DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
that you can trust, that you can depend on, that is worthy of your
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
```

```
bridge is a bridge that doesn't crack even under hea
sending ACK[2]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=876
        DATA packet blocknum = 2
vy load, and a
reliable transistor is one that operates for years, possibly under
extreme temperature and radiation.

The common enemy of reliability in these examples are adverse
circumstances and influences that may cause a deterioration of the
physical properties of material.  The accumulation of these influences
is called aging.  Reliability is achieved by dimensioning the
mechanisms properly, taking such adverse conditions into
consideration.  In a railway system the schedule is arranged such that
it l
sending ACK[3]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=0
        DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
that you can trust, that you can depend on, that is worthy of your
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[2]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=1148
        DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
that you can trust, that you can depend on, that is worthy of your
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
```

```
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[2]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=2400
        DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
that you can trust, that you can depend on, that is worthy of your
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[2]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=0
        DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
that you can trust, that you can depend on, that is worthy of your
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[2]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=199
        DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
that you can trust, that you can depend on, that is worthy of your
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
```

```
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[2]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=1055
        DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
that you can trust, that you can depend on, that is worthy of your
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[2]
hard timeout
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=3546
        DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
that you can trust, that you can depend on, that is worthy of your
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[2]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=0
        DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
that you can trust, that you can depend on, that is worthy of your
confidence.  For example, a reliable clock is one that indicates
```

```
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[2]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=733
        DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
that you can trust, that you can depend on, that is worthy of your
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[2]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=836
        DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
that you can trust, that you can depend on, that is worthy of your
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[2]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=1045
        DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
that you can trust, that you can depend on, that is worthy of your
confidence.  For example, a reliable clock is one that indicates
```

```
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[2]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=1043
        DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
that you can trust, that you can depend on, that is worthy of your
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[2]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=1044
        DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
that you can trust, that you can depend on, that is worthy of your
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[2]
hard timeout
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=5556
        DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
that you can trust, that you can depend on, that is worthy of your
```

```
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[2]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=0
        DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
that you can trust, that you can depend on, that is worthy of your
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[2]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=0
        DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
that you can trust, that you can depend on, that is worthy of your
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[2]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=1
        DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
that you can trust, that you can depend on, that is worthy of your
```

```
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[2]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=571
        DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
that you can trust, that you can depend on, that is worthy of your
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[2]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=1075
        DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
that you can trust, that you can depend on, that is worthy of your
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[2]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=1043
        DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
that you can trust, that you can depend on, that is worthy of your
```

```
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[2]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=1044
        DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
that you can trust, that you can depend on, that is worthy of your
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[2]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=1007
        DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
that you can trust, that you can depend on, that is worthy of your
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[2]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=1081
        DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
that you can trust, that you can depend on, that is worthy of your
```

```
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[2]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=1044
        DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
that you can trust, that you can depend on, that is worthy of your
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[2]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=998
        DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
that you can trust, that you can depend on, that is worthy of your
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[2]
hard timeout
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=5469
        DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
```

```
that you can trust, that you can depend on, that is worthy of your
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[2]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=0
        DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
that you can trust, that you can depend on, that is worthy of your
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[2]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=0
        DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
that you can trust, that you can depend on, that is worthy of your
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[2]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=0
        DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
```

```
that you can trust, that you can depend on, that is worthy of your
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[2]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=632
	DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
that you can trust, that you can depend on, that is worthy of your
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[2]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=1043
	DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
that you can trust, that you can depend on, that is worthy of your
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[2]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=1044
	DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
```

```
that you can trust, that you can depend on, that is worthy of your
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[2]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=1044
	DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
that you can trust, that you can depend on, that is worthy of your
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[2]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=999
	DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
that you can trust, that you can depend on, that is worthy of your
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[2]
hard timeout
hard timeout
hard timeout
hard timeout
```

Delay

```
Last login: Fri Dec  8 17:09:09 on ttys002
Yangs-MacBook-Air:~ yangli$ clear; cd "/Users/yangli/Desktop/COMP443/YangLiWUMPProject/YangLiwumpt/bin/Debug";
"/Library/Frameworks/Mono.framework/Versions/5.2.0/bin/mono32"  --debug --debugger-
agent=transport=dt_socket,address=127.0.0.1:63282  "/Users/yangli/Desktop/COMP443/YangLiWUMPProject/YangLiwump
t/bin/Debug/wumpt1.exe" ; echo $? > /var/folders/24/2mdjgs2x76zfx9554t61_bxw0000gn/T/tmp6cc4fe31.tmp; echo; read -p
'Press any key to continue...' -n1; exit
```

```
Looking up address of ulam1.cs.luc.edu... got it: 147.126.2.91
req size = 10, filename=delay
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=2
        DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
that you can trust, that you can depend on, that is worthy of your
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
```

```
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=0
        DATA packet blocknum = 2
vy load, and a
reliable transistor is one that operates for years, possibly under
extreme temperature and radiation.

The common enemy of reliability in these examples are adverse
circumstances and influences that may cause a deterioration of the
physical properties of material.  The accumulation of these influences
is called aging.  Reliability is achieved by dimensioning the
mechanisms properly, taking such adverse conditions into
consideration.  In a railway system the schedule is arranged such that
it l
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=1
        DATA packet blocknum = 3
eaves room for catching up on lost time, and an ample supply of
spare engines is kept on the alert for emergencies.  A bridge is built
stronger than actually needed most of the time -- and a transistor is
equipped with cooling devices and radiation shields.

What does all this have to do with software?  Well, we all have
experienced failures of compter systems; and we all would like them to
be reliable too.  When a computer fails, the first question among its
intimates is usually: is the hardware or the sof
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=434
        DATA packet blocknum = 2
vy load, and a
reliable transistor is one that operates for years, possibly under
extreme temperature and radiation.

The common enemy of reliability in these examples are adverse
circumstances and influences that may cause a deterioration of the
physical properties of material.  The accumulation of these influences
is called aging.  Reliability is achieved by dimensioning the
mechanisms properly, taking such adverse conditions into
consideration.  In a railway system the schedule is arranged such that
```

```
it l
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=365
        DATA packet blocknum = 2
vy load, and a
reliable transistor is one that operates for years, possibly under
extreme temperature and radiation.

The common enemy of reliability in these examples are adverse
circumstances and influences that may cause a deterioration of the
physical properties of material.  The accumulation of these influences
is called aging.  Reliability is achieved by dimensioning the
mechanisms properly, taking such adverse conditions into
consideration.  In a railway system the schedule is arranged such that
it l
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=469
        DATA packet blocknum = 2
vy load, and a
reliable transistor is one that operates for years, possibly under
extreme temperature and radiation.

The common enemy of reliability in these examples are adverse
circumstances and influences that may cause a deterioration of the
physical properties of material.  The accumulation of these influences
is called aging.  Reliability is achieved by dimensioning the
mechanisms properly, taking such adverse conditions into
consideration.  In a railway system the schedule is arranged such that
it l
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=417
        DATA packet blocknum = 2
vy load, and a
reliable transistor is one that operates for years, possibly under
extreme temperature and radiation.

The common enemy of reliability in these examples are adverse
circumstances and influences that may cause a deterioration of the
physical properties of material.  The accumulation of these influences
```

is called aging. Reliability is achieved by dimensioning the mechanisms properly, taking such adverse conditions into consideration. In a railway system the schedule is arranged such that
it l

```
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=417
	DATA packet blocknum = 2
```

vy load, and a
reliable transistor is one that operates for years, possibly under extreme temperature and radiation.

The common enemy of reliability in these examples are adverse circumstances and influences that may cause a deterioration of the physical properties of material. The accumulation of these influences is called aging. Reliability is achieved by dimensioning the mechanisms properly, taking such adverse conditions into consideration. In a railway system the schedule is arranged such that
it l

```
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=312
	DATA packet blocknum = 2
```

vy load, and a
reliable transistor is one that operates for years, possibly under extreme temperature and radiation.

The common enemy of reliability in these examples are adverse circumstances and influences that may cause a deterioration of the physical properties of material. The accumulation of these influences is called aging. Reliability is achieved by dimensioning the mechanisms properly, taking such adverse conditions into consideration. In a railway system the schedule is arranged such that
it l

```
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=418
	DATA packet blocknum = 2
```

vy load, and a
reliable transistor is one that operates for years, possibly under extreme temperature and radiation.

The common enemy of reliability in these examples are adverse circumstances and influences that may cause a deterioration of the physical properties of material. The accumulation of these influences is called aging. Reliability is achieved by dimensioning the mechanisms properly, taking such adverse conditions into consideration. In a railway system the schedule is arranged such that it l

```
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=417
        DATA packet blocknum = 2
```

vy load, and a
reliable transistor is one that operates for years, possibly under extreme temperature and radiation.

The common enemy of reliability in these examples are adverse circumstances and influences that may cause a deterioration of the physical properties of material. The accumulation of these influences is called aging. Reliability is achieved by dimensioning the mechanisms properly, taking such adverse conditions into consideration. In a railway system the schedule is arranged such that it l

```
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=417
        DATA packet blocknum = 2
```

vy load, and a
reliable transistor is one that operates for years, possibly under extreme temperature and radiation.

The common enemy of reliability in these examples are adverse circumstances and influences that may cause a deterioration of the physical properties of material. The accumulation of these influences is called aging. Reliability is achieved by dimensioning the mechanisms properly, taking such adverse conditions into consideration. In a railway system the schedule is arranged such that it l

```
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=417
        DATA packet blocknum = 2
```

vy load, and a

reliable transistor is one that operates for years, possibly under extreme temperature and radiation.

The common enemy of reliability in these examples are adverse circumstances and influences that may cause a deterioration of the physical properties of material. The accumulation of these influences is called aging. Reliability is achieved by dimensioning the mechanisms properly, taking such adverse conditions into consideration. In a railway system the schedule is arranged such that
it l

```
sending ACK[1]
hard timeout
hard timeout
hard timeout
hard timeout
```

Spray

(Expected_block++)

```
Last login: Fri Dec  8 17:10:56 on ttys000
Yangs-MacBook-Air:~ yangli$ clear; cd "/Users/yangli/Desktop/COMP443/YangLiWUMPProject/YangLiwumpt/bin/Debug";
"/Library/Frameworks/Mono.framework/Versions/5.2.0/bin/mono32"  --debug --debugger-
agent=transport=dt_socket,address=127.0.0.1:63513  "/Users/yangli/Desktop/COMP443/YangLiWUMPProject/YangLiwump
t/bin/Debug/wumpt1.exe" ; echo $? > /var/folders/24/2mdjgs2x76zfx9554t61_bxw0000gn/T/tmp7ea05a6c.tmp; echo; read -p
'Press any key to continue...' -n1; exit
```

```
Looking up address of ulam1.cs.luc.edu... got it: 147.126.2.91
req size = 10, filename=delay
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=3
        DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
that you can trust, that you can depend on, that is worthy of your
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[2]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=0
        DATA packet blocknum = 2
vy load, and a
reliable transistor is one that operates for years, possibly under
extreme temperature and radiation.

The common enemy of reliability in these examples are adverse
circumstances and influences that may cause a deterioration of the
physical properties of material.  The accumulation of these influences
is called aging.  Reliability is achieved by dimensioning the
mechanisms properly, taking such adverse conditions into
consideration.  In a railway system the schedule is arranged such that
it l
sending ACK[3]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=1
        DATA packet blocknum = 3
eaves room for catching up on lost time, and an ample supply of
```

spare engines is kept on the alert for emergencies. A bridge is built stronger than actually needed most of the time -- and a transistor is equipped with cooling devices and radiation shields.

What does all this have to do with software? Well, we all have experienced failures of compter systems; and we all would like them to be reliable too. When a computer fails, the first question among its intimates is usually: is the hardware or the sof

```
sending ACK[4]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=0
        DATA packet blocknum = 4
```

tware the culprit?
Most customers of a computation center show signs of relief when the latter is announced, for the disruption of service is then quickly ended by a so-called deadstart, and life goes on as if (almost) nothing had occurred. Indeed there had been neither an earthquake, nor a snowstorm, nor a weighty load, nor heat or radiation. Instead, merely unpredictable circumstances had led to a state of computation for which the logical structure of the program had not been designed, which the system

```
sending ACK[5]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=1
        DATA packet blocknum = 5
```

's designers didn't anticipate. And when pressing the deadstart button, the computer operator is reasonably confident that these circumstances won't reoccur too soon.

What must we conclude? We understand by the term software the collection of programs that deterministically prescribe a system's detailed behaviour and transtions of state. These programs are constants and are independent of any "adverse conditions" of an environment. Hence, software cannot fail because of unpredictable happenings and age,

```
sending ACK[6]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=0
        DATA packet blocknum = 6
```

 but only due to defects in its logical design. This
leads us to a replacement of the attribute "reliable" by "correct."

We may be accused of nitpicking with words. To this I can only reply that the choice of words often reveals a speaker's attitude more profoundly than is clear to him. The attitude through which we content ourselves at producing "reliable" software instead of correct software, bears the danger that we may also consider various degrees of reliability. Software may then be termed reliabl

```
sending ACK[7]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=1
        DATA packet blocknum = 7
```

e and "more reliable";
we may also call it correct, but certainly not "more correct."

The difference in these words is also manifested in the techniques to be employed in producing reliability in software versus in clocks, bridges, and transistors. In most technical phenomena, reliability is achieved by overdimensioning the components, by using high quality material, or by supplying standby equipment that automatically goes into action when a failure occurs. In programs, merely repeating a logical test t

```
sending ACK[8]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=0
        DATA packet blocknum = 8
```

en times instead of performing it once does not help, if the logical structure is correct and the underlying hardware is reliable. In fact, the degree to which a program is unreliable is exactly the probability with which its incorrect parts are executed. But this measure is not a property of the program itself.

Reconciling ourselves with the word correct in place of reliable has the advantage that we more readily identify the causes of failures of our products to meet their goal. They are not to be soug

```
sending ACK[9]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=1
        DATA packet blocknum = 9
```

ht in
external, unforeseeable, adverse circumstances, but solely in our own inadequate minds, and in our failure to communicate, if several people participate in a program's design. The advantage of this recognition is that we know where to concentrate our efforts; its unpleasant part

```
is the fact that it will be a neverending crusade, because committing
mistakes is a truly human characteristic.

The most sensible targets in our drive at producing correct software
are evidently the programmers themselves.
sending ACK[10]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=0
        DATA packet blocknum = 10
Nothing whatsoever can
replace a sound, systematic training in precise reasoning.  Other
sensible targets are the tools that we employ to assist our reasoning.
These include primarily the formal languages in which we express our
thoughts and abstractions, and by which we transmit them to other
people.  We have directed our efforts to improve our programming tools
since more than a decade, and I will therefore devote the main part of
this paper to a report and an evaluation of the latest product, the
languag
sending ACK[11]
rec'd packet: len=66; proto=1; opcode=2; src=(147.126.2.91/4517); time=1
        DATA packet blocknum = 11
e Pascal, in the light of the topic "reliable software."



sending ACK[12]
hard timeout
hard timeout
hard timeout
hard timeout
```

Reorder

```
Last login: Fri Dec  8 17:11:59 on ttys000
Yangs-MacBook-Air:~ yangli$ clear; cd "/Users/yangli/Desktop/COMP443/YangLiWUMPProject/YangLiwumpt/bin/Debug"; "/Library/Frameworks/Mono.framework/Versions/5.2.0/bin/mono32"  --debug --debugger-agent=transport=dt_socket,address=127.0.0.1:63674  "/Users/yangli/Desktop/COMP443/YangLiWUMPProject/YangLiwumpt/bin/Debug/wumpt1.exe" ; echo $? > /var/folders/24/2mdjgs2x76zfx9554t61_bxw0000gn/T/tmp1d49faf8.tmp; echo; read -p 'Press any key to continue...' -n1; exit
```

```
Looking up address of ulam1.cs.luc.edu... got it: 147.126.2.91
req size = 12, filename=reorder
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=2
        DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
that you can trust, that you can depend on, that is worthy of your
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=0
        DATA packet blocknum = 2
vy load, and a
reliable transistor is one that operates for years, possibly under
extreme temperature and radiation.

The common enemy of reliability in these examples are adverse
circumstances and influences that may cause a deterioration of the
```

physical properties of material. The accumulation of these influences is called aging. Reliability is achieved by dimensioning the mechanisms properly, taking such adverse conditions into consideration. In a railway system the schedule is arranged such that
it l

```
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=2
        DATA packet blocknum = 3
```

eaves room for catching up on lost time, and an ample supply of spare engines is kept on the alert for emergencies. A bridge is built stronger than actually needed most of the time -- and a transistor is equipped with cooling devices and radiation shields.

What does all this have to do with software? Well, we all have experienced failures of compter systems; and we all would like them to be reliable too. When a computer fails, the first question among its intimates is usually: is the hardware or the sof

```
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=401
        DATA packet blocknum = 2
```

vy load, and a
reliable transistor is one that operates for years, possibly under extreme temperature and radiation.

The common enemy of reliability in these examples are adverse circumstances and influences that may cause a deterioration of the physical properties of material. The accumulation of these influences is called aging. Reliability is achieved by dimensioning the mechanisms properly, taking such adverse conditions into consideration. In a railway system the schedule is arranged such that
it l

```
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=1426
        DATA packet blocknum = 2
```

vy load, and a
reliable transistor is one that operates for years, possibly under extreme temperature and radiation.

The common enemy of reliability in these examples are adverse circumstances and influences that may cause a deterioration of the physical properties of material. The accumulation of these influences is called aging. Reliability is achieved by dimensioning the mechanisms properly, taking such adverse conditions into consideration. In a railway system the schedule is arranged such that it l

```
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=0
        DATA packet blocknum = 2
```

vy load, and a reliable transistor is one that operates for years, possibly under extreme temperature and radiation.

The common enemy of reliability in these examples are adverse circumstances and influences that may cause a deterioration of the physical properties of material. The accumulation of these influences is called aging. Reliability is achieved by dimensioning the mechanisms properly, taking such adverse conditions into consideration. In a railway system the schedule is arranged such that it l

```
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=588
        DATA packet blocknum = 2
```

vy load, and a reliable transistor is one that operates for years, possibly under extreme temperature and radiation.

The common enemy of reliability in these examples are adverse circumstances and influences that may cause a deterioration of the physical properties of material. The accumulation of these influences is called aging. Reliability is achieved by dimensioning the mechanisms properly, taking such adverse conditions into consideration. In a railway system the schedule is arranged such that it l

```
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=213
        DATA packet blocknum = 2
```

vy load, and a reliable transistor is one that operates for years, possibly under

```
extreme temperature and radiation.

The common enemy of reliability in these examples are adverse
circumstances and influences that may cause a deterioration of the
physical properties of material.  The accumulation of these influences
is called aging.  Reliability is achieved by dimensioning the
mechanisms properly, taking such adverse conditions into
consideration.  In a railway system the schedule is arranged such that
it l
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=401
        DATA packet blocknum = 2
vy load, and a
reliable transistor is one that operates for years, possibly under
extreme temperature and radiation.

The common enemy of reliability in these examples are adverse
circumstances and influences that may cause a deterioration of the
physical properties of material.  The accumulation of these influences
is called aging.  Reliability is achieved by dimensioning the
mechanisms properly, taking such adverse conditions into
consideration.  In a railway system the schedule is arranged such that
it l
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=429
        DATA packet blocknum = 2
vy load, and a
reliable transistor is one that operates for years, possibly under
extreme temperature and radiation.

The common enemy of reliability in these examples are adverse
circumstances and influences that may cause a deterioration of the
physical properties of material.  The accumulation of these influences
is called aging.  Reliability is achieved by dimensioning the
mechanisms properly, taking such adverse conditions into
consideration.  In a railway system the schedule is arranged such that
it l
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=418
```

```
        DATA packet blocknum = 2
vy load, and a
reliable transistor is one that operates for years, possibly under
extreme temperature and radiation.

The common enemy of reliability in these examples are adverse
circumstances and influences that may cause a deterioration of the
physical properties of material.  The accumulation of these influences
is called aging.  Reliability is achieved by dimensioning the
mechanisms properly, taking such adverse conditions into
consideration.  In a railway system the schedule is arranged such that
it l
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=521
        DATA packet blocknum = 2
vy load, and a
reliable transistor is one that operates for years, possibly under
extreme temperature and radiation.

The common enemy of reliability in these examples are adverse
circumstances and influences that may cause a deterioration of the
physical properties of material.  The accumulation of these influences
is called aging.  Reliability is achieved by dimensioning the
mechanisms properly, taking such adverse conditions into
consideration.  In a railway system the schedule is arranged such that
it l
sending ACK[1]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=313
        DATA packet blocknum = 2
vy load, and a
reliable transistor is one that operates for years, possibly under
extreme temperature and radiation.

The common enemy of reliability in these examples are adverse
circumstances and influences that may cause a deterioration of the
physical properties of material.  The accumulation of these influences
is called aging.  Reliability is achieved by dimensioning the
mechanisms properly, taking such adverse conditions into
consideration.  In a railway system the schedule is arranged such that
```

```
it l
sending ACK[1]
hard timeout
hard timeout
```

Reorder

(Expected_block++)

```
Last login: Fri Dec  8 17:12:43 on ttys000
Yangs-MacBook-Air:~ yangli$ clear; cd "/Users/yangli/Desktop/COMP443/YangLiWUMPProject/YangLiwumpt/bin/Debug";
"/Library/Frameworks/Mono.framework/Versions/5.2.0/bin/mono32"  --debug --debugger-
agent=transport=dt_socket,address=127.0.0.1:63908  "/Users/yangli/Desktop/COMP443/YangLiWUMPProject/YangLiwump
t/bin/Debug/wumpt1.exe" ; echo $? > /var/folders/24/2mdjgs2x76zfx9554t61_bxw0000gn/T/tmp4dded1cf.tmp; echo; read -p
'Press any key to continue...' -n1; exit
```

```
Looking up address of ulam1.cs.luc.edu... got it: 147.126.2.91
```

```
req size = 12, filename=reorder
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=2
        DATA packet blocknum = 1
An Assessment of the Programming Language Pascal
by Niklaus Wirth, developer of Pascal

1. What is reliable software?
Reliable is the attribute for a person, an organization, or a mechanism
that you can trust, that you can depend on, that is worthy of your
confidence.  For example, a reliable clock is one that indicates
accurate time even during an earthquake, a reliable railway system is
one where trains run punctually even during a snowstorm, a reliable
bridge is a bridge that doesn't crack even under hea
sending ACK[2]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=0
        DATA packet blocknum = 2
vy load, and a
reliable transistor is one that operates for years, possibly under
extreme temperature and radiation.

The common enemy of reliability in these examples are adverse
circumstances and influences that may cause a deterioration of the
physical properties of material.  The accumulation of these influences
is called aging.  Reliability is achieved by dimensioning the
mechanisms properly, taking such adverse conditions into
consideration.  In a railway system the schedule is arranged such that
it l
sending ACK[3]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=1
        DATA packet blocknum = 3
eaves room for catching up on lost time, and an ample supply of
spare engines is kept on the alert for emergencies.  A bridge is built
stronger than actually needed most of the time -- and a transistor is
equipped with cooling devices and radiation shields.

What does all this have to do with software?  Well, we all have
experienced failures of compter systems; and we all would like them to
be reliable too.  When a computer fails, the first question among its
intimates is usually: is the hardware or the sof
```

```
sending ACK[4]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=0
        DATA packet blocknum = 4
tware the culprit?
Most customers of a computation center show signs of relief when the
latter is announced, for the disruption of service is then quickly
ended by a so-called deadstart, and life goes on as if (almost) nothing
had occurred.  Indeed there had been neither an earthquake, nor a
snowstorm, nor a weighty load, nor heat or radiation.  Instead, merely
unpredictable circumstances had led to a state of computation for which
the logical structure of the program had not been designed, which the
system
sending ACK[5]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=1
        DATA packet blocknum = 5
's designers didn't anticipate.  And when pressing the deadstart
button, the computer operator is reasonably confident that these
circumstances won't reoccur too soon.

What must we conclude? We understand by the term software the
collection of programs that deterministically prescribe a system's
detailed behaviour and transtions of state.  These programs are
constants and are independent of any "adverse conditions" of an
environment.  Hence, software cannot fail because of unpredictable
happenings and age,
sending ACK[6]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=0
        DATA packet blocknum = 6
 but only due to defects in its logical design. This
leads us to a replacement of the attribute "reliable" by "correct."

We may be accused of nitpicking with words.  To this I can only reply
that the choice of words often reveals a speaker's attitude more
profoundly than is clear to him.  The attitude through which we content
ourselves at producing "reliable" software instead of correct software,
bears the danger that we may also consider various degrees of
reliability.  Software may then be termed reliabl
sending ACK[7]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=1
```

```
        DATA packet blocknum = 7
e and "more reliable";
we may also call it correct, but certainly not "more correct."

The difference in these words is also manifested in the techniques to
be employed in producing reliability in software versus in clocks,
bridges, and transistors.  In most technical phenomena, reliability is
achieved by overdimensioning the components, by using high quality
material, or by supplying standby equipment that automatically goes
into action when a failure occurs.  In programs, merely repeating a
logical test t
sending ACK[8]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=0
        DATA packet blocknum = 8
en times instead of performing it once does not help, if
the logical structure is correct and the underlying hardware is
reliable.  In fact, the degree to which a program is unreliable is
exactly the probability with which its incorrect parts are executed.
But this measure is not a property of the program itself.

Reconciling ourselves with the word correct in place of reliable has
the advantage that we more readily identify the causes of failures of
our products to meet their goal.  They are not to be soug
sending ACK[9]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=1
        DATA packet blocknum = 9
ht in
external, unforeseeable, adverse circumstances, but solely in our own
inadequate minds, and in our failure to communicate, if several people
participate in a program's design.  The advantage of this recognition
is that we know where to concentrate our efforts; its unpleasant part
is the fact that it will be a neverending crusade, because committing
mistakes is a truly human characteristic.

The most sensible targets in our drive at producing correct software
are evidently the programmers themselves.
sending ACK[10]
rec'd packet: len=520; proto=1; opcode=2; src=(147.126.2.91/4517); time=0
        DATA packet blocknum = 10
```

Nothing whatsoever can replace a sound, systematic training in precise reasoning. Other sensible targets are the tools that we employ to assist our reasoning. These include primarily the formal languages in which we express our thoughts and abstractions, and by which we transmit them to other people. We have directed our efforts to improve our programming tools since more than a decade, and I will therefore devote the main part of this paper to a report and an evaluation of the latest product, the languag

```
sending ACK[11]
rec'd packet: len=66; proto=1; opcode=2; src=(147.126.2.91/4517); time=1
        DATA packet blocknum = 11
```

e Pascal, in the light of the topic "reliable software."

```
sending ACK[12]
hard timeout
hard timeout
```