Issues Insufficiently Resolved in Century 20 in the Fault-Tolerant Distributed Computing Field

Kane Kim
UCI DREAM Lab
khkim@uci.edu, http://dream.eng.uci.edu/

For Invited Presentation at
SRDS 2000, Nuremberg
Oct. 2000

Outline

• Liveliness of the FT DC field
• FT DC advances in Century 20
• Issues insufficiently resolved
Issues Insufficiently Resolved in Century 20 ----
- An Ill-Chosen Subject ? (1)

- Too big an area
Issues Insufficiently Resolved in Century 20 - An Ill-Chosen Subject? (2)

- Politically incorrect
  - Exposing one’s ignorance and biases

* Moreover, ∃ chilling statements which are the opposite of the cheerleaders’ statements.
Issues Insufficiently Resolved in Century 20 ----
- An Ill-Chosen Subject ？ (2)

• Politically incorrect
  - Exposing one’s ignorance and biases

* Moreover, ∃ chilling statements which are the opposite of the
  cheerleaders’ statements.

? Invitation to speak at SRDS
  = Order to retire from active research ?

Oct-00 | 4
Liveness of the FT DC Field

- Industry specializing in fault-tolerant (FT) computing never flourished.
  - The reliability of hardware components continuously improved at a spectacular pace.
  - DBMS vendors successfully built the data backup and simple transaction mechanisms.
  - But system software support needed for higher-coverage FT computing (e.g., automatic retry of a failed transaction following system reconfiguration) did not advance into mature forms.

- The interests of main-stream computing industry was mostly confined to facilitating clean abort of transactions whenever faults in intermediate computation results or uncommitted data were found.

- The FT distributed computing (DC) field is bouncing back up again due to:
  1. Rapid growth of the Web server market and customers' growing demands for high-availability Web servers, and
  2. Rapid growth of RT computing applications that started around mid-1990's, especially growing demands for computer-embedded communication-equipped devices / systems in this new decade.
Liveness of the FT DC Field
-Growing Motivations for Higher-Coverage FT DC Approaches

Web server

* End users lose patience when Web sites handling competitive commercial activities take longer than 8 seconds to show results.

=>

(1) Such Web sites must be up "all the time",
i.e., meet high-availability requirements; and

(2) Web sites must respond fast even when they are accessed by a large number of clients concurrently.

=>

High-coverage FT DC
Liveness of the FT DC Field
-Growing Motivations for Higher-Coverage FT DC Approaches

RT Computing Applications

* No longer a negligible market even for major platform vendors

• Mere clean abort of transactions leads more often than not to abandoning the application.

• Video-conferencing, voice over IP (internet protocol), factory automation, defense applications, etc., require higher coverage in FT DC than the Web server application does.

=> Attempts for automated retry of a failed transaction or concurrent redundant tries of a transaction are usually essential.

I.E., Forward or backward RT recovery from faults is a usual attempt.
Liveness of the FT DC Field
-Evolution of the interests of main-stream computing industry

Clean Abort

High Availability Server

RT Recovery
FT DC Advances in Century 20

- I. Hardened Hardware Component Technologies

* Spectacular advances in integration of numerous logic components

• Significant advances in late 1970’s and 1980’s in producing specially hardened hardware modules

(1) Hardened processor modules:

Comparing processor-pair, pair of self-checking processors, and voting-TMR (triple modular redundancy) processor module.

(2) RAID (redundant array of inexpensive disks):

Popular even in database-centric business computing applications.

(3) Error-detection and error-correction coding subunit:

Extensively used in CPUs and communication processors and various peripheral devices.

• Standard general-purpose hardware modules have become quite reliable and powerful in performance

=> Growing interests in using software techniques
Various basic approaches were established.

1. **Timeout**: mature
2. **Comparison** of the results of repeated or redundant executions; mature
3. **Error-detection and error-correction code**: mature
4. **Acceptance test**: Test reasonableness of intermediate computation results
**FT DC Advances in Century 20**

- **11. Fault Detection and Network Surveillance**

* Various basic approaches were established.

(1) **Timeout**; *mature*

(2) **Comparison** of the results of repeated or redundant executions; *mature*

(3) **Error-detection and error-correction code**; *mature*

(4) **Acceptance test**: Test reasonableness of intermediate computation results

(5) **Network surveillance**, also called **membership maintenance**:
   - Simplest version: Master node make a periodic roll-call of other nodes
   - Yet a small number of techniques which are practical and also yield to rigorous quantitative analyses of fault coverage
     - The periodic reception history broadcast (PRHB) scheme and the time-triggered protocol (TTP) scheme for use in bus-LAN based systems
     - The supervisor-based network surveillance (SNS) scheme for use in point-to-point network based systems

* **Important metric**: Detection-latency bound
FT DC Advances in Century 20
- III. Transaction

• Established by the DB research community
  - Based on the notion of atomicity and *sphere of control* formulated earlier

• Aimed for maintaining
  atomicity,
  consistency,
  isolation, and
  durability

  in spite of component failures

* Just doing clean abort meets these requirements.

• Log-based schemes for efficient abort and commit and
  schemes for concurrent execution of multiple transactions
  have been well developed.
FT DC Advances in Century 20
- IV. Checkpointing and Recovery Lines

• Rollback-retry, also called checkpointing-recovery, was a technique developed in 1960's to increase the probability of successful completion of a sequential atomic real-time computation-segment.

• A recovery line for a process, say P1, is:

  A set of checkpoints, each belonging to a different process, which will not be crossed by a rollback of any process caused by the failure of P1.

• Recovery lines have been extensively studied for the past 25 years but how much acceptance by practitioners?
**FT DC Advances in Century 20**

- V. Replication

- **Replicated databases** have been extensively studied in the past 3 decades
  - The simplest type where every transaction is executed on the replica designed as the primary and a subsequent update command is sent to other replicas, has been the most popular.

- **Replicated processes**:
  - There are 6 basic types
  - **FT computing station** = a combination of replicated processes and executing node facilities

**Structure 1:** Comparing pair and rollback
FT DC Advances in Century 20
- V. Replication

Structure 2: Pair of self-checking Processing nodes (PSP)

Structure 2a: Pair of single-processor nodes with an application-independent fault detection software component
Uses the primary-shadow cooperation scheme

Structure 2b: Pair of comparing pairs (PCP)

Diagram:
- Hardware / Software Checker

[Diagram of hardware/software checker]
Structure 3: Distributed Recovery Block (DRB) Station

Predecessor Computing Station

Input Buffer

Logical & Time

AT

Time

AT

F

S

Input data

"Data ID"

"I'm OK"

Output OK

Initial Primary Node X

Initial Shadow Node Y

Successor Computing Station

Primary

Shadow

State saving
• Advantages of DRB
  • **Forward recovery** in the same manner regardless of whether a node fails due to **hardware faults** or **software faults**;
  • The **increase in the normal task turnaround time** is minimal (because the primary node does not wait for any status message from the shadow node);
  • The **cost-effectiveness** and the **flexibility** are high because
    c1) a DRB computing station can operate **with just two try blocks and two processing nodes** and
    c2) the two try blocks are not required to produce identical results and the second try block need not be as sophisticated as the first try block.

• If **software fault tolerance** is not a goal, alternate algorithms are not needed and providing acceptance tests is also optional.
  => The DRB structure becomes the PSP Structure.
Structure 4: Voting triple modular redundant (TMR) station
(or more generally, N-modular redundant station)

Structure 5: N-Version Programming (NVP) station
* multiple versions expected to generate truly identical computation results -- restrictive
Issues Insufficiently Resolved
- I. Quantitative Treatment

- Effective, let alone optimal, resource allocation is not possible in the absence of quantitative characterizations of FT schemes.
  - Yet the research efforts made are grossly inadequate.
Issues Insufficiently Resolved
- 1. Quantitative Treatment

• Effective, let alone optimal, resource allocation is not possible in the absence of quantitative characterizations of FT schemes.
  – Yet the research efforts made are grossly inadequate.

Most important metrics

• Where clean abort is the recovery goal:
  (1) Fault types and rates covered,
  (2) The extra hardware costs, and
  (3) The extra time costs
    (incl. overhead for enabling fault detection, abortion time, and server-down time).

• Where RT recovery is desirable:
  (1) Fault types and rates covered ;
  (2) Recovery time bound :
    Maximum difference between a normal task execution time and the time for a task execution involving fault detection and recovery events ;
  * In some applications such as space exploration, extra hardware costs
Issues Insufficiently Resolved
- I. Quantitative Treatment

- FT approaches not yielding to easy quantitative analyses are unsafe to use.

- Ideally,
  
  analyzable node OS,
  analyzable middleware, and
  analyzable application software

must be used to realize reliable DC systems.

Adding FT capabilities cannot be an excuse for violating this law.
Issues Insufficiently Resolved
- 1. Quantitative Treatment

Fair and Unfair Modeling of Fault Sources

- Fault source model of a DC system
  \[ \text{:= A combination (or an abstraction of a combination) of the faulty behavior models of components used in composing the system} \]

- Unfortunately, more often than not,
  Subjective and non-scientific reasoning was used in adopting and assessing the reasonableness of fault source models.

- This often led to the lack of harmony, the lack of trust, and the lack of open-minded spirits among the researchers in the FT DC field.
Issues Insufficiently Resolved
- I. Quantitative Treatment

Fair and Unfair Modeling of Fault Sources (cont)

- A good fault source model must be a characterization of all "non-negligible" patterns of fault occurrences.

- Replaceable components most often modeled as:
  - Fail-silent unit (FSU):
    - can exhibit only absence of an explicit output upon occurrence of any internal fault.
    - Idealistic component model
    - Too simplistic in some situations
  - Malicious unit (MaU), also called the Byzantine unit
    - Mostly hard to justify
**Idealistic Direction**: Develop a systematic method for fair distribution of concerns over possible occurrences of anomalous events during system design and validation.

- E.g., Malicious fault
- Fair Distribution
- Unfair Distribution

E.g., # (Faulty nodes) = N/2
Fair distribution of concerns over possible occurrences of anomalous events

- (FDR1) All non-negligible events which the designer can envision are placed within the encircled space,

- (FDR2) No events which are placed outside the circular boundary and thus to be ignored by designers and/or evaluators, have occurrence probabilities which are more than several magnitudes-of-order (e.g., 10 - 100 times) greater than the occurrence probability of any event placed within the encircled space.
Issues Insufficiently Resolved
- 1. Quantitative Treatment

Server-down time and Recovery time bound

• Important research topic
Intended output actions of RT computations always take place on time in spite of fault occurrences.

- If not feasible, the fault tolerance actions which lead to the least damages to the application missions / users must be attempted.

To be practically useful,

- Fault detection technique must at least yield a tightly bounded detection latency and
- Recovery technique must at least yield a tightly bounded recovery time.

∃ signs that even the major vendors of OS and communication infrastructure are gradually stepping up their efforts in making the timing behavior of their products more predictable.
Main Challenge: Integration

- RT FT computing stations + Network surveillance and reconfiguration (NSR):
  - To improve fault coverage and detection latency and recovery time bounds

- Fault detection and replication principles + Object-oriented (OO) RT DC structuring techniques:
  - RT OO programming movement is a cutting-edge technology movement initiated in 1990's.
  - Goal of that movement: Instigate a quantum productivity jump in software engineering for RT DC application systems.
  - Adapting the existing RT fault tolerance techniques for integration into the powerful RT OO DC structure is an important challenge.

Ref. WORDS
Issues Insufficiently Resolved
- II. RT FT DC

Scalability

- Growing sizes of applications and increasing use of WANs
- Time-based coordination of distributed actions (pioneered by Kopetz) is a fundamental approach insufficiently explored.
Issues Insufficiently Resolved
- III. Reliable Multicast

- Group communication without fault tolerance is a trivial application of single point-to-point message communications.

- Sensible to cast FT group communication protocols as
  - Distributed application programs supported by
  - Execution engines using established RT fault tolerance techniques.
    - Should be capable of effectively handling failures of low-level components such as
      processors,
      paths in communication / interconnection networks,
      processor-network interfaces, and
      OS components.

  Rather than the other way around

- Challenge:
  - Realize a tight bound on the multicast time under a reasonable fault source model
Issues Insufficiently Resolved
- IV. OO FT DC

• **OO DC movement**, e.g., CORBA movement, Java-based DC movement, DCOM and SOAP movement by Microsoft, etc., has become a major technological movement.

• **FT OO DC** technology is becoming an active R&D field.

• **Challenge:** Exploitation of *intra-object concurrency* while enabling *high-coverage FT computing* such as RT recovery.
Issues Insufficiently Resolved  
- V. Software FT

• The most difficult research issue
  - Research community has become tiny.
  - These brave researchers should be encouraged.

• Challenge:
  
  Show a convincing demo.
  
  Use of artificially injected faults will not be a fully valid approach.
Summary

- The liveliness of the FT DC field is in an upward move at this opening juncture of Century 21.

- Major holes in the established foundation:
  - Quantitative characterizations of FT DC techniques,
  - Enhancement needed in RT FT DC technologies, especially,
    - integration of RT FT computing station construction techniques and network surveillance and reconfiguration (NSR) techniques, and
    - application of established fault detection and replication principles to OO RT DC structuring techniques, and
  - OO FT DC, and
  - Software fault tolerance.
Summary (cont)

• Pays off to understand the technical foundation established in Century 20.

Randell and Dobson in SRDS 1986:

"As a profession, we seem to specialise in re-inventing the wheel, and in inventing jargon that, by accident or design, obscures the fact of re-invention."

Summary (cont)

- Future emphasis on **quantitative treatment** of design techniques & protocols and **scientific assessment of fault source models** will lead to accelerated advances in FT DC technologies.
Summary (cont)

- Future emphasis on quantitative treatment of design techniques & protocols and scientific assessment of fault source models will lead to accelerated advances in FT DC technologies.

- It will also hopefully lead to more efficient, open-minded, unemotional reasoning atmosphere, which will have better effect of encouraging young researchers to enter and stay in the field.