Reliability

Reliability, failure & tolerance

Faults during different phases of design

- Inconsistent or inadequate specifications
- Program errors
- Component & communication system failures
- User-level exception handling required to handle the failure

Failure modes

- Non-termination / -completion
- Transient faults
- Program errors
- Range violations and other inconsistent states
- Value violations and other wrong results

Value domain

- Early error omission
- Faults of a certain regularity demand careful analysis
- Faults which stay in the system continue to cause problems
- Un-manned vehicles which operate semi-autonomously by default
- Systems in remote / hostile environments

Achieving reliability

System identification

- Static applications specifications
- Physical sensors and converters constraints
- Constraints of the employed control network
- Constraints of the underlying run-time system

Investigate:

- Use reliable hardware components — Consider the environmental demands!
- Physical sensors and converters constraints.
- Constraints of the employed controller network.
- Constraints of the underlying run-time system.

To understand all critical real-time requirements and issues.

Fault avoidance

- Full fault avoidance
- Select the means to operate the novel system, such as reliable hardware components, hardware redundancy, software redundancy, or a combination thereof.

Configuration of the system

- For the configuration of the system, the system designer must ensure that:
  - The system meets all requirements.
  - The system is reliable.

Fault tolerance

- Full fault tolerance
- Partial fault tolerance
- Fault-tolerance mechanisms
- Fault-tolerance mechanisms in practice

Hardware redundancy

- Adding extra hardware resources
- N-Modular Redundancy (NMR)
- N-Modular Redundancy (NMR)
- N-Modular Redundancy (NMR)

The assumption that errors occur in a repeatable manner is difficult to justify. Therefore, the system designer must ensure that:
- The system meets all requirements.
- The system is reliable.

Reliability

Reliability, failure & tolerance

Terminology of failure or ‘failing terminology’

Reliability — a measure of success with which a system conforms to its specification.

Failure — a deviation of a system from its specification.

Error — the system state which leads to a failure.

Fault — there exists an error for an error.
The six-language project

The six-language project is an initiative between the USA Department of Defense and the six major software contractors that developed the software for the F-22 Raptor and F-35 Joint Strike Fighter. 

The specifications include detailed and accurate original source descriptions, documentation, and design models; additional certification points; certification and validation activities; and page-sized diagrams.

All communication and information is required to follow predefined protocols and must be approved by designated authorities before any new information is introduced. The information must be validated by the corresponding authorities.

The project has a timeline of $\text{day} \times \text{hours} \times \text{minutes}$ for the development cycle.


table

<table>
<thead>
<tr>
<th>N-version programming – Voting issues</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integer arithmetic:</td>
</tr>
<tr>
<td>- Image areas in a given input: These results will be divided.</td>
</tr>
<tr>
<td>Real arithmetic:</td>
</tr>
<tr>
<td>- Real-world results will be divided. Computations need to consider tolerances, as slight differences in real runtime could cause significant errors.</td>
</tr>
</tbody>
</table>

Multiple solutions:

- Error recovery: error recovery algorithms on multiple test, structurally different solutions.
- X: opens the system.

Fault treatment

Fault treatment involves adjusting the system to withstand failures. Typically, this involves the following steps:

1. Error detection: Detecting and authorizing error conditions.
2. Error correction: Adjusting the system to correct errors.
3. Error avoidance: Preventing errors from occurring in the first place.
4. Error recovery: Reacting to errors and returning the system to a normal state.

Dynamic redundancy

Dynamic redundancy is a method of providing multiple, structurally different solutions. The basic idea is to reduce the error rate by providing multiple, independent computations. If one computation fails, another can be used to provide the correct result.

Failure categories

There are several categories of failures that can occur in complex systems. These include:

- Single failure: A single failure that causes a system to fail.
- Multiple failures: Two or more failures that cause a system to fail.
- System failure: A failure that causes the entire system to fail.

Impact of errors

Error recovery: Error recovery methods can be categorized as:

- Hardware: Using hardware to detect and correct errors.
- Software: Using software to detect and correct errors.
- Error logging: Recording errors for future analysis.

Advantages of dynamic redundancy include:

- Reduced error rates: By using multiple, independent computations, the error rate can be significantly reduced.
- Increased system reliability: The system can continue to operate even if one or more of the computations fail.
- Improved fault tolerance: The system can tolerate a wide range of errors and still continue to operate.

Disadvantages of dynamic redundancy include:

- Increased cost: The cost of implementing dynamic redundancy can be high due to the additional hardware and software required.
- Increased complexity: The system becomes more complex due to the additional error detection and recovery mechanisms.
- Increased power consumption: Dynamic redundancy can increase the power consumption of the system.

Application areas

Dynamic redundancy is particularly useful in safety-critical systems, such as aerospace and defense systems, where the failure of a single component can have serious consequences.

Example

A typical example of dynamic redundancy is the implementation of a trip-wire system in an aircraft. In this system, multiple redundant trip-wires are used to detect and correct errors. If one trip-wire fails, another can be used to provide the correct result.

Further reading

For more information on dynamic redundancy, please refer to the following resources:

- [Reference 1], [Reference 2], [Reference 3].
### Ada Ravenscar profile

- **Task type and object declarations at the library level**
  - No hierarchy of tasks, and hence no exit protocols needed from blocks and sub-programs.
  - No dynamic allocation or unchecked de-allocation of protected and task objects.
- **Restrict**
  - No requeue.
  - No use of dynamic priorities.
  - No abort or ATC.
- **Library level Protected objects**
  - With a Count attribute.
  - Atomic and Volatile pragmas: these provide atomic updates to shared data and can be implemented simply.
  - Necessary to enforce the correct use of shared data.
- **Ada Certification profiles**
  - Profiles tend to specific certification processes, e.g., DO-178B.
  - Managed by the Radio Technical Commission for Aeronautics (RTCA).
  - Addressed e.g., Model driven design, verification, formal methods.
  - Components of the software will be distinguished.
  - No safety of crit. minor, major, rescue, catastrophic.

### Metric-Interval Temporal Logic

- Assumptions over time events:
  - Implicitly, logical and numeric simulation.
- WRN: discontinuity at the intersection of event classes.

### Linear Temporal Logic of Real Numbers (LTR)

- Omit assumptions of predicates.