Applying Reliability Models to the Space Shuttle

Norman F. Schneidewind

Ted W. Keller

What are Reliability Models?

Reliability Models are tools that Help Us:

- **Predict** software Reliability.
- **Control** software Reliability.
- **Assess** software Reliability.

This functions let an organization determine if the reliability goals it sets for its software has been met.

Evaluating Reliability Models

- We usually need to evaluate candidate reliability models and select the models that best match the software’s failure history.

- The US Space Shuttle is a case study on how a real project team did that.

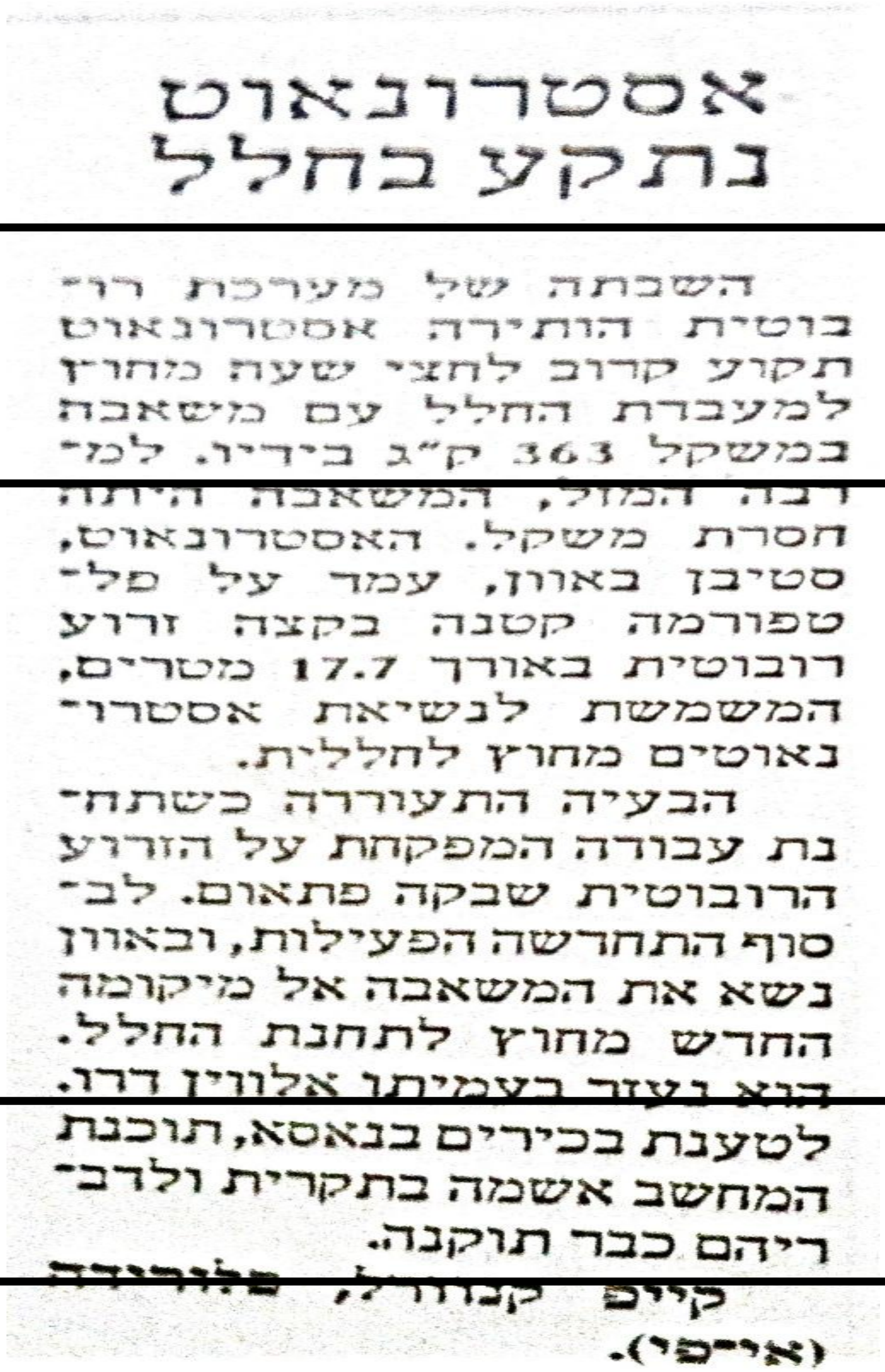
The US Space Shuttle

The IBM Federal Services Company in Huston, selected the **Shneidewind model** to predict the reliability of the shuttle's on-board system software for NASA, After evaluating many reliability models and tried to validate them for use on this project.

We will see how.

System Failure

“A failure is the inability of a system or system component to perform a required function within specified limits.” (IEE Standard Glossary of Software Engineering Terminology, New York,1983)



אסטרונאוט נתקע בחלל

השבתה של מערכת רובוטית הותירה אסטרונאוט תקוע קרוב לחצי שעה מחוץ למעבדת החלל עם משאבה במשקל 363 ק"ג בידיו. למרבה המזל, המשאבה היתה חסרת משקל. האסטרונאוט, סטיבן באוון, עמד על פלטפורמה קטנה בקצה זרוע רובוטית באורך 17.7 מטרים, המשמשת לנשיאת אסטרונאוטים מחוץ לחללית.

הבעיה התעוררה כשתחנת עבודה המפקחת על הזרוע הרובוטית שבקה פתאום. לבסוף התחדשה הפעילות, ובאוון נשא את המשאבה אל מיקומה החדש מחוץ לתחנת החלל. הוא נעזר בעמיתו אלווין דרו. לטענת בכירים בנאסא, תוכנת המחשב אשמה בתקרית ולדבריהם כבר תוקנה.

קייפ קנוורל, פלורידה (איי־פי).

Reliable Program

3 separate but related functions comprise an integrated reliability program:

- **Prediction**- Estimating the future failure rate, number of failures, time to next failure, mean time to failure.

  We use statistical modeling in order to predict reliability.

- **Control**- Comparing predictions with predefined goals and flagging software that fails to meet those goals.

- **Assessment** – What action to take when software fails to meet goals.

  Assessment also includes formulating test strategies (talked later on)

Schneidewind model assumptions

- A system is modified only in immediate response to an observed failure.

- The process used to correct the code is constant.

- All code in a program is homogeneous from the stand point of execution history.

Schneidewind model assumptions - con

These assumptions appear at first to represent significant incompatibilities for many systems.

To apply your data to a reliability model, consider breaking your system and processes into smaller elements that can more accurately satisfy assumptions.

Shuttle's Primary Avionics Software Sub System (PASS)

The Shuttle's Primary Avionics Software Sub System is modified frequently using a constantly improving process to add or change capabilities.

More then 15 version of PASS have been released to NASA since 1980, each an upgrade of the preceding version.

Satisfying Assumptions

The IBM team used the “breaking your system” approach we discussed, to deal with the Schneidwind model’s assumption.

Let’s look on how they did it on PASS (next slide).

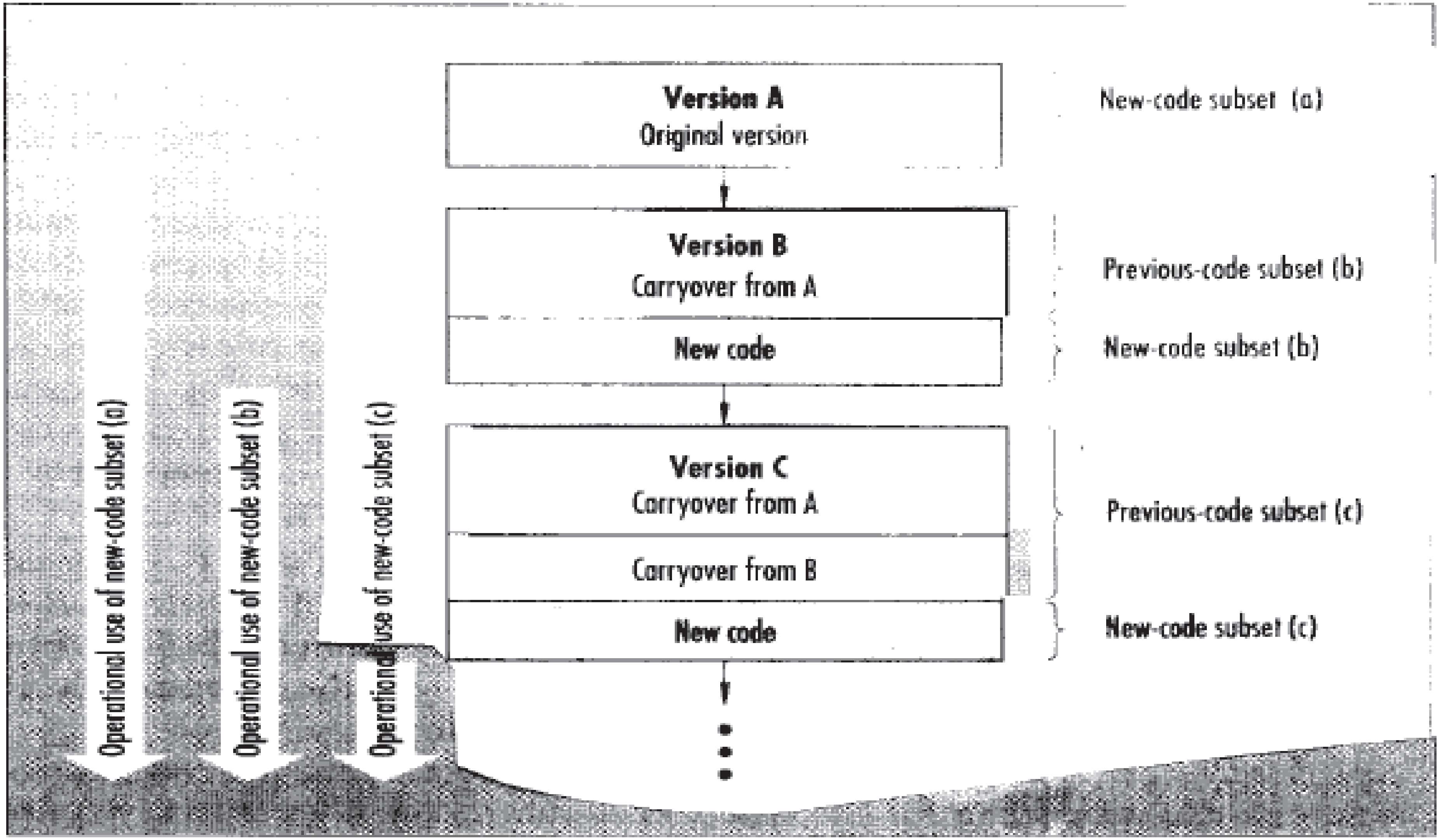


Figure 1. *Depiction of a sequentially upgraded system. The first release is the original version, labeled version A. All the code in version A is new, so subset a is the entire system. The white arrows show when each new-code subset begins operation. When the system is updated and rereleased as version B, only the lines of code that were modified or added are included in new-code subset b. All of version B is carried over to version C, unless it is modified, in which case it becomes part of new-code subset c. This process applies to each successive system release.*

Validation

- The IBM team selected several models for evaluation, on a history of 100 failures.
- The team used the failure data of six dates between 1986 and 1989 to obtain six PASS reliability predictions.
- The Scheidwind model’s reliability predictions were about 15 percent less than the actual average time between failures, and it appears to provide the most accurate fit to the 12 years of failure data.

Reliability and Testing

- If you don’t have a testing strategy, test costs are likely to get out of control.

- You must treat modules unequally.

Allocate more test time , funds and effort to modules with the highest predicted number of failures.

Reliability and Testing – con.

- You can use reliability model to predict failures, F(t1,t2), during the interval t1,t2 where t could be execution time or tester labor time for a single module (in this case, t means execution time).

- The article’s recommendation is to allocate test time to modules in proportion to F(t1,t2).

Reliability and Testing – con.

$X_{0,t1}$ Is the actual number of fails during (0,t1).

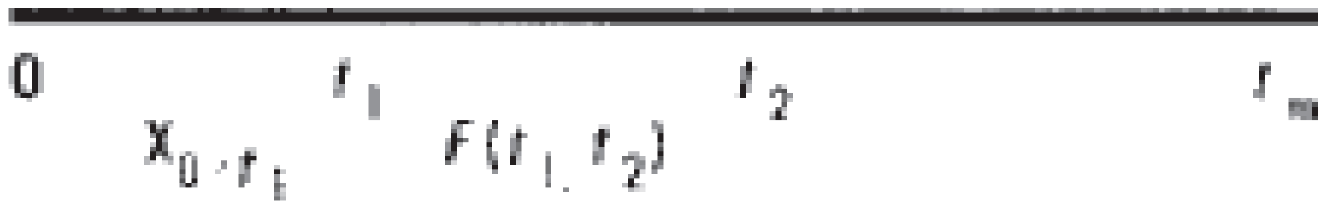


Figure 2. Reliability–prediction time scale. We predict $F(t_1,t_2)$, at t_1 during the time interval t_1,t_2, based on the model and X_{0,t_1}. On the time scale, t_m is total available test time for a single module.

Equations

$$F(t) = (\alpha/\beta)[1 - \exp(-\beta t)] \quad (1)$$

$$F(t_1,t_2) = (\alpha/\beta)[1 - \exp(-\beta t_2)] - X_{0,t_1} \quad (2)$$

$$F(\infty) = \alpha/\beta \quad (3)$$

$$R(t) = (\alpha/\beta) - X_{0,t} \quad (4)$$

$$T_i = \frac{F_i(t_1,t_2) * (n)\,[t_2 - t_1]}{\sum_{i=1}^{n} F_i(t_1,t_2)} \quad (5)$$

Example

- The team used the interval 0,20 to estimate **α** and **β** for each module and the interval 20,30 to make predictions for each module.

- They calculated T_i for each module, which is dependent on prediction results.

- Units are days.

TABLE 1
OBSERVED FAILURES AND PARAMETERS FOR 0,20 INTERVAL

	Failures	α	β
Module 1	12	1.6915	.1306
Module 2	11	1.7642	.1411
Module 3	10	1.7464	.1615

TABLE 2
TEST-RESOURCE ALLOCATION

	$F(\infty)$ failures	F(20,30) failures	R(20) failures	T periods	X(20, 20+T) failures
Module 1					
predicted	12.95	.695	.952	7.6	
actual	13	0	1		0
Module 2					
predicted	12.50	1.322	1.503	14.4	
actual	13	1	2		1
Module 3					
predicted	10.81	.729	.814	8.0	
actual	14	1	4		1

When to stop testing?

The actual $F(\infty)$

Is known only after all testing is complete.

- You need additional procedures in for deciding how long to test to reach a given number of remaining failures.

When to stop testing? – con.

The writers recommended approach to deciding when to stop testing uses reliability prediction to estimate the minimum testing time t2 (or the interval (0,t2) needed to reduce the predicted maximum number of remaining failures R(t2).

$R(t_2)$ can be established from

$$R(t_2) = (p)(\alpha/\beta)$$

where p is the desired fraction of remaining failures at t_2.

$$t_2 = \{\ln [(1/p)]\}/\beta$$

TABLE 3
TIME NEEDED TO REACH "ZERO" REMAINING FAILURES
(p = .001)

	Total test time (in periods)	Additional test time (in periods)	Time to find last failure (in periods)
Module 1	52.9	45.3	64
Module 2	49.0	34.6	44

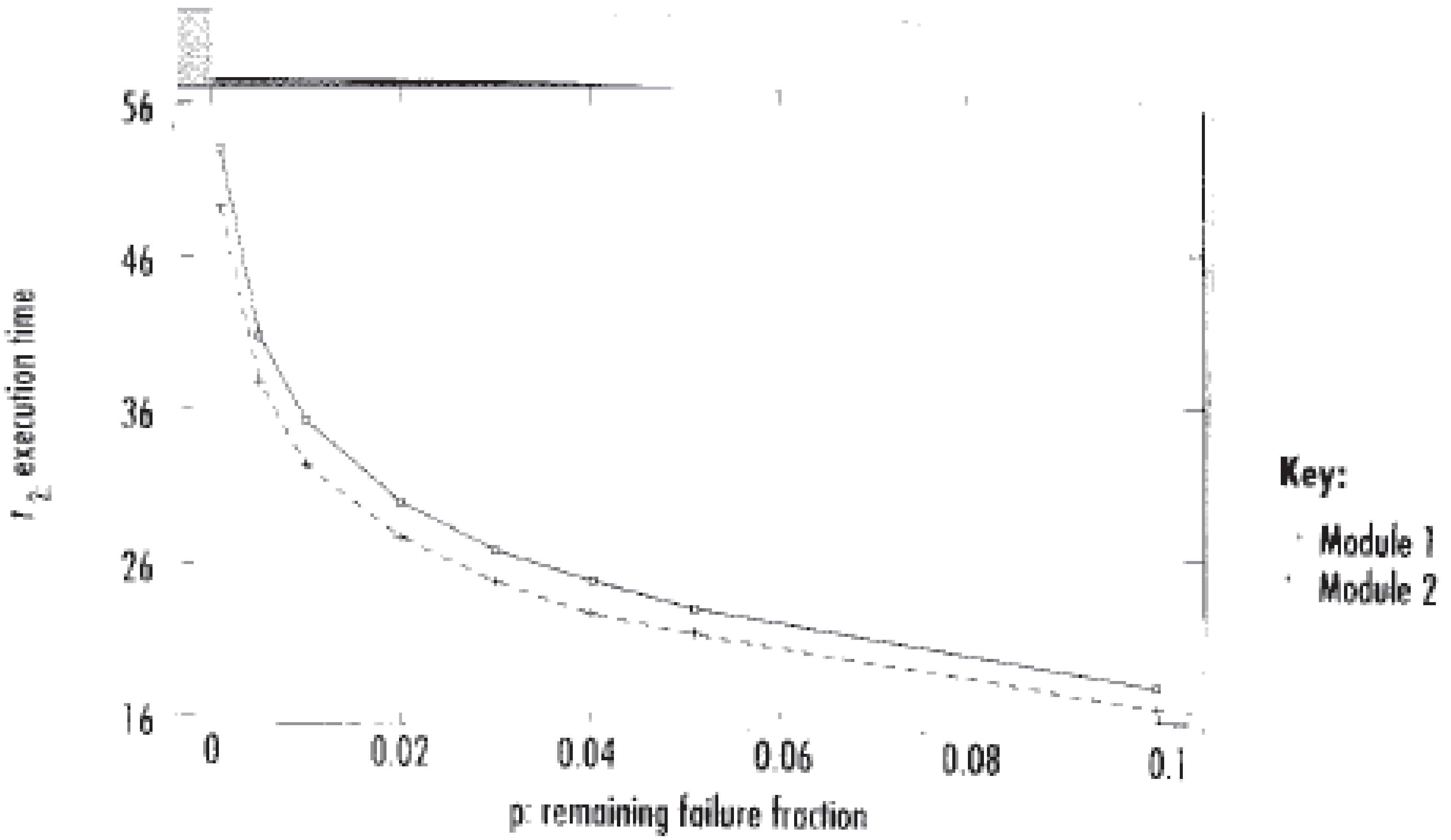


Figure 3. Execution time needed to reach the desired fraction of remaining failures.