Emc[2]
Innovative Structural Integrity Solutions

Extremely Low Probability of Rupture Code: Lessons learned from 15 years of development and applications

Cédric Sallaberry, Robert Kurth

Frederick Brust, Elizabeth Kurth-Twombly

Markus Burkardt

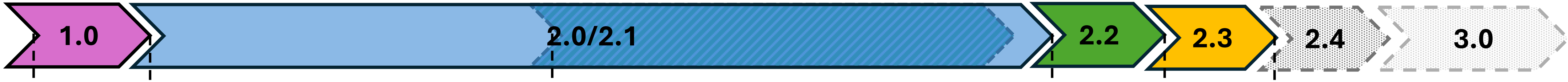
Dominion Engineering, Inc.

Nathan Glunt

EPRI
ELECTRIC POWER RESEARCH INSTITUTE

xLPR Timeline

- xLPR development spans over 15 years and continues



2008/2009
Beginning of the project

2010
Version 1.0 “Pilot version” completed (internal release)

2015
Version 2.0 completed (internal release)

April 2020
xLPR v2.1 publicly released

March 2022
xLPR v2.2 released

August 2024
xLPR v2.3 released

Pilot Study

Beneficial:

- Smaller scope: Faster to develop and save time as some errors were identified early on and fixed when developing the next version.
- Lessons learned reported in NUREG-2110 (# ML12145A470).
- Better understanding of the overall goal for the whole development team, which allowed moving to a top-down approach.

Needed Improvement:

- Reliance on the pilot study. Simpler assumptions were made, but some of the new development was still based on those assumptions (e.g., crack numbering by subsegment location).

Multi Science-centric approach

Emc² Innovative Structural Integrity Solutions

Beneficial:

- Having experts in fracture mechanics and in probabilistic analysis from the start. Ensure that all aspects of PFM code are considered:
  - FE modelers for Weld Residual Stresses confirmed that 3rd order or 4th order polynomial fit would not work for some profiles. Keep the physics realistic. And WRS influences the results A LOT.
  - Having risk analyst/statistician from the start helped the top-down development and the definition of the quantities of interest.
- Relying on existing knowledge and previous codes (save development time)
- Involvement from both the regulators and vendors/utilities brings convergence toward conservative but still realistic approaches.

Needed Improvement:

- Lack of software programmers. Software development is a full-time job now. Scientists still at the core of the module, but expert developers needed for efficient and modern programming.

Modular approach

Beneficial:

- Initial goal was to give the user the possibility to develop their own module/equations and “plug” them to the framework. **Still valid point.**
- Having each mechanism in a separate module lead to smoother evolution. Modules are dissociated from the framework and do not need to be updated at the same time.
- xLPR was developed as a larger PFM platform so that it can cover other degradation mechanisms and component configurations in the future.

Needed Improvement:

- Not enough effort was given for the configuration. xLPR is currently limited for crack evolution in welded pipes.
- Some requirements were identified later during the development and needed some specific implementation (pre-processor).
- Code has a high requirement in running time and memory, limiting the estimate of extremely rare events (less than 10^{-6}) with SRS **(not the only reason)**.

QA + Verification and Validation

Beneficial:

- Large and consistent development of test-cases at the module level AND the framework level. Testing how each module perform by itself and how all perform together as a complex code. Several errors were found and corrected.
- Extensive documentation for each module, for the framework and for the inputs.

Needed Improvement:

- The purpose of each QA document was not clear and led to confusion and inconsistencies when writing them (equations in SRD or SDD or both?).
- Cost of maintenance is high.
- Bugs/errors continue to be identified after completion of V&V.

Training

Beneficial:

- Extensive training material, with examples to run, recorded videos, large user manual.

Needed Improvement:

- Some of the training material (older videos) is outdated.
- Maintenance cost of training is high.
- Still steep learning curve for new users. The code needs an initial knowledge in the physics involved and in probabilistic approach: even when running a deterministic example.
- The development of a user group was considered to support new users, but it requires a logistic effort and enough users.

Code purpose and life

A software remains alive as long as it has a use

2008 Vision
Comprehensive, vetted, adaptable, and flexible

- Reliance on Pilot Study
- Lack of software programmers
- Inconsistency in development (preprocessor)
- **Memory and Time limitations**
- Limited problem configuration
- Cost of maintenance of QA and V&V
- Cost of maintenance of training

- Steep Learning Curve

2023 Vision
Simple, efficient, flexible, high-quality tool for PFM analyses in rulemaking, design, and aging management

- Development of V3.0 framework with software programmers from top down again
- New optimization modules
- Identifying new areas of applications
- Automatisms in testing and documents generation (GitHub environment, doorstop...) and training examples

- You're supposed to be smart, so you're on your own for that.