



More than 25 Years of CalculiX: Sustaining an Industrial Open-Source FEM Code

22.07.2026 | L. Bruder, C. Wölfe, G. Dhondt

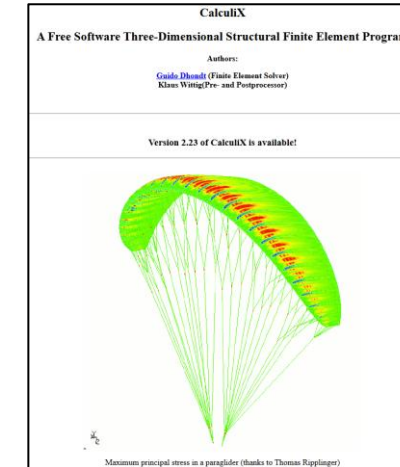
CalculiX – a Multidisciplinary 3D Finite Element Software

Started in 1998 in spare time

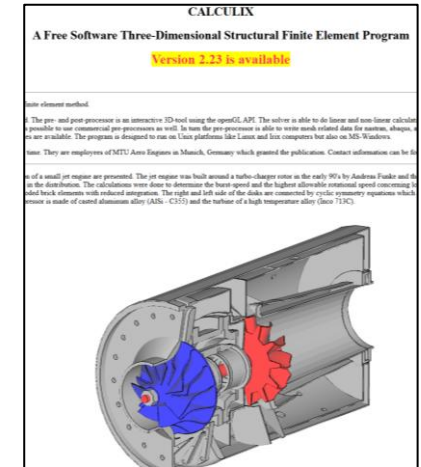
- CalculiX CrunchiX → Solver (Guido Dhondt) → [this talk](#)
- CalculiX GraphiX → Pre- and Postprocessor (Klaus Wittig)
- ABAQUS-like .inp input format (supports a limited set of keywords)
- .frd / .dat output format

Since 2000

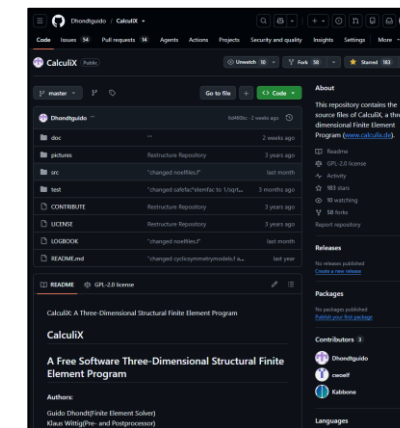
- Distributed under the GPL-2.0 license
- Continuous feature enrichment
- Co-developers (mostly students & university partners) for specific implementations
- Increasing usage at MTU (especially structural mechanics, heat transfer and secondary air system)
 - Reduced license costs (massive job parallelization / important for multi-query tasks such as optimization)
 - Custom feature development
 - Strengthening in-house FEM know-how
- Increasing user community (mostly structural mechanics & heat transfer)



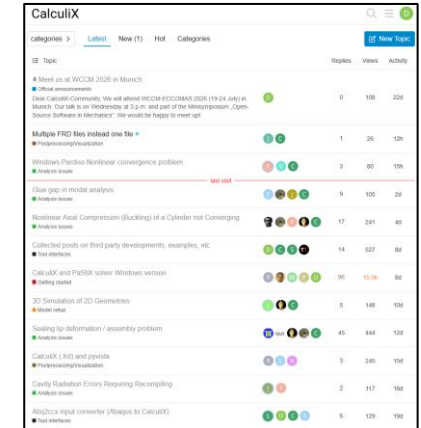
dthondt.de



calculix.de



github.com/Dhondtguido/CalculiX



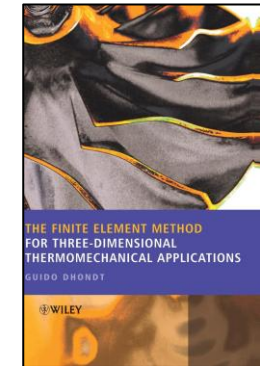
https://calculix.discourse.group

CalculiX – a Multidisciplinary 3D Finite Element Software

- CalculiX User's Manual (> 800 pages)
- Dhondt, G. The Finite Element Method for Three-Dimensional Thermomechanical Applications, Wiley, 2004. (→ new issue in preparation)
- Several publications (journal articles, student thesis, etc.)
- Test suite with > 600 minimal examples

CalculiX CrunchiX USER'S MANUAL version 2.23
Guido Dhondt
October 19, 2025

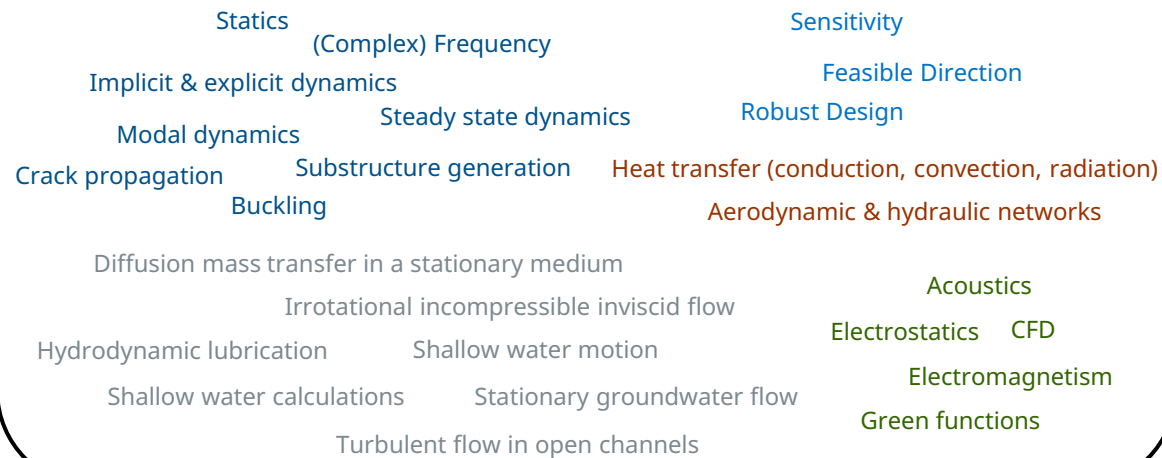
Contents	
1. Introduction	12
2. How to perform CalculiX calculations in parallel	12
3. Units	15
4. Golden rules	16
5. Simple example problems	19
5.1. Cantilever beam	19
5.2. Frequency calculation of a beam loaded by a compressive force	20
5.3. Frequency calculation of a rotating disk on a plastic shell	29
5.4. Thermal expansion of a beam	31
5.5. Beam under a load	31
5.6. Continuity of a composite beam	31
5.7. Hydrostatic stress	35
5.8. Adhesive joint	36
5.9. Thermal expansion in a composite beam	36
5.10. Stresses in a beam under a compressive force	36
5.11. Stresses in a beam under a compressive force	36
5.12. Stresses in a beam under a compressive force	36
5.13. Stresses in a beam under a compressive force	36
5.14. Stresses in a beam under a compressive force	36
5.15. Stresses in a beam under a compressive force	36
5.16. Stresses in a beam under a compressive force	36
5.17. Stresses in a beam under a compressive force	36
5.18. Stresses in a beam under a compressive force	36
5.19. Stresses in a beam under a compressive force	36
5.20. Stresses in a beam under a compressive force	36
5.21. Stresses in a beam under a compressive force	36
5.22. Stresses in a beam under a compressive force	36
5.23. Stresses in a beam under a compressive force	36
5.24. Stresses in a beam under a compressive force	36
5.25. Stresses in a beam under a compressive force	36
5.26. Stresses in a beam under a compressive force	36
5.27. Stresses in a beam under a compressive force	36
5.28. Stresses in a beam under a compressive force	36
5.29. Stresses in a beam under a compressive force	36
5.30. Stresses in a beam under a compressive force	36
5.31. Stresses in a beam under a compressive force	36
5.32. Stresses in a beam under a compressive force	36
5.33. Stresses in a beam under a compressive force	36
5.34. Stresses in a beam under a compressive force	36
5.35. Stresses in a beam under a compressive force	36
5.36. Stresses in a beam under a compressive force	36
5.37. Stresses in a beam under a compressive force	36
5.38. Stresses in a beam under a compressive force	36
5.39. Stresses in a beam under a compressive force	36
5.40. Stresses in a beam under a compressive force	36
5.41. Stresses in a beam under a compressive force	36
5.42. Stresses in a beam under a compressive force	36
5.43. Stresses in a beam under a compressive force	36
5.44. Stresses in a beam under a compressive force	36
5.45. Stresses in a beam under a compressive force	36
5.46. Stresses in a beam under a compressive force	36
5.47. Stresses in a beam under a compressive force	36
5.48. Stresses in a beam under a compressive force	36
5.49. Stresses in a beam under a compressive force	36
5.50. Stresses in a beam under a compressive force	36
5.51. Stresses in a beam under a compressive force	36
5.52. Stresses in a beam under a compressive force	36
5.53. Stresses in a beam under a compressive force	36
5.54. Stresses in a beam under a compressive force	36
5.55. Stresses in a beam under a compressive force	36
5.56. Stresses in a beam under a compressive force	36
5.57. Stresses in a beam under a compressive force	36
5.58. Stresses in a beam under a compressive force	36
5.59. Stresses in a beam under a compressive force	36
5.60. Stresses in a beam under a compressive force	36
5.61. Stresses in a beam under a compressive force	36
5.62. Stresses in a beam under a compressive force	36
5.63. Stresses in a beam under a compressive force	36
5.64. Stresses in a beam under a compressive force	36
5.65. Stresses in a beam under a compressive force	36
5.66. Stresses in a beam under a compressive force	36
5.67. Stresses in a beam under a compressive force	36
5.68. Stresses in a beam under a compressive force	36
5.69. Stresses in a beam under a compressive force	36
5.70. Stresses in a beam under a compressive force	36
5.71. Stresses in a beam under a compressive force	36
5.72. Stresses in a beam under a compressive force	36
5.73. Stresses in a beam under a compressive force	36
5.74. Stresses in a beam under a compressive force	36
5.75. Stresses in a beam under a compressive force	36
5.76. Stresses in a beam under a compressive force	36
5.77. Stresses in a beam under a compressive force	36
5.78. Stresses in a beam under a compressive force	36
5.79. Stresses in a beam under a compressive force	36
5.80. Stresses in a beam under a compressive force	36
5.81. Stresses in a beam under a compressive force	36
5.82. Stresses in a beam under a compressive force	36
5.83. Stresses in a beam under a compressive force	36
5.84. Stresses in a beam under a compressive force	36
5.85. Stresses in a beam under a compressive force	36
5.86. Stresses in a beam under a compressive force	36
5.87. Stresses in a beam under a compressive force	36
5.88. Stresses in a beam under a compressive force	36
5.89. Stresses in a beam under a compressive force	36
5.90. Stresses in a beam under a compressive force	36
5.91. Stresses in a beam under a compressive force	36
5.92. Stresses in a beam under a compressive force	36
5.93. Stresses in a beam under a compressive force	36
5.94. Stresses in a beam under a compressive force	36
5.95. Stresses in a beam under a compressive force	36
5.96. Stresses in a beam under a compressive force	36
5.97. Stresses in a beam under a compressive force	36
5.98. Stresses in a beam under a compressive force	36
5.99. Stresses in a beam under a compressive force	36
5.100. Stresses in a beam under a compressive force	36



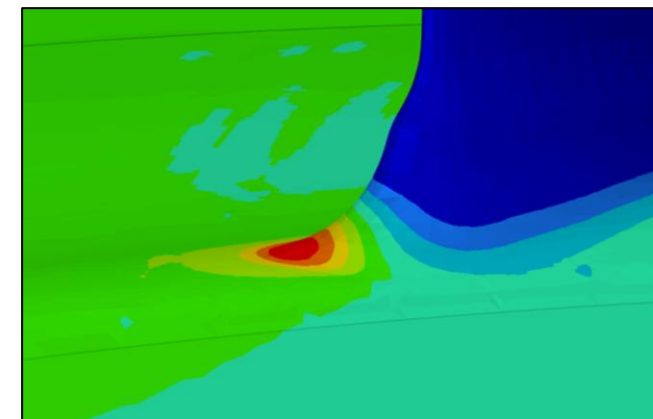
User's Manual

Book

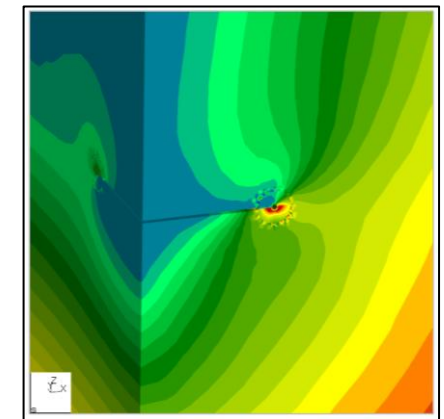
Supported types of analysis



Custom proprietary developments based on the CalculiX solver



Shape optimization via vertex morphing

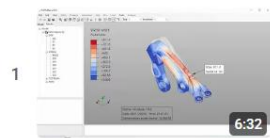


Cracktracer3D

CalculiX Community

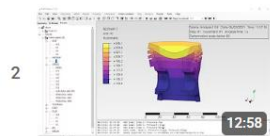
Several initiatives (mostly decentralized / independent)

- Dedicated pre- and postprocessors: [CalculiX GraphiX](#), [PrePoMax](#)
- Software interfacing with CalculiX: [preCICE](#), [FreeCAD](#), [Mecway](#)
- GitHub projects: [ccx2paraview](#), [Cubit-CalculiX](#), [abq2ccx](#), [CalculiX-Examples](#), ...
- Packaged by Linux distributions
- Tutorials & discussions
- Validated against ISO 10211-2017 (Thermal bridges in building construction)
- Used by other companies



PrePoMax & CalculiX - Basic Tutorial

Matej Borovinšek • 36K views • 8 years ago



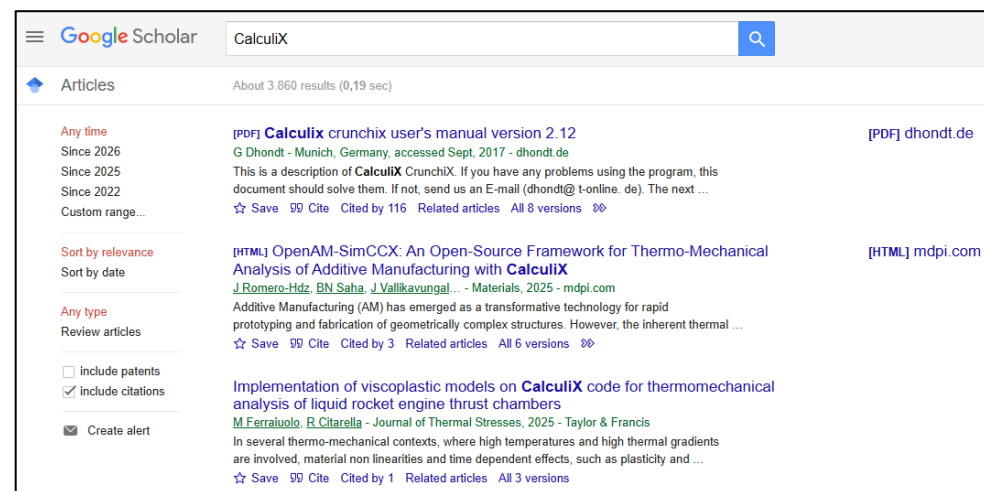
PrePoMax & CalculiX - Coupled thermo-mechanical

Matej Borovinšek • 7.7K views • 4 years ago



Statistics on community activity

- Site downloads per month
 - ~1000 source code (latest version)
 - ~300 Linux executable (latest version)
 - ~400 Windows executable (latest version)
- ~800 registered users on Discourse with ~100 posts per month (+ a lot more anonymous readers)
- ~3800 hits on Google Scholar



Development model

Status quo for a long time

- Single author development
- Code distribution via website
- Bug reporting via email
- Contributions via email

Pros

- Mature and stable software
- Consistent coding style and documentation
- Most-used functionality well tested and optimized

Cons

- Very few community developers
- Redistribution of marginally different custom binaries / sources

Transitioning to a team-based development

- Development team at MTU
- Allocating responsibilities
- Establishing a git strategy
- Issue tracking
- CI/CD



Encouraging community contributions

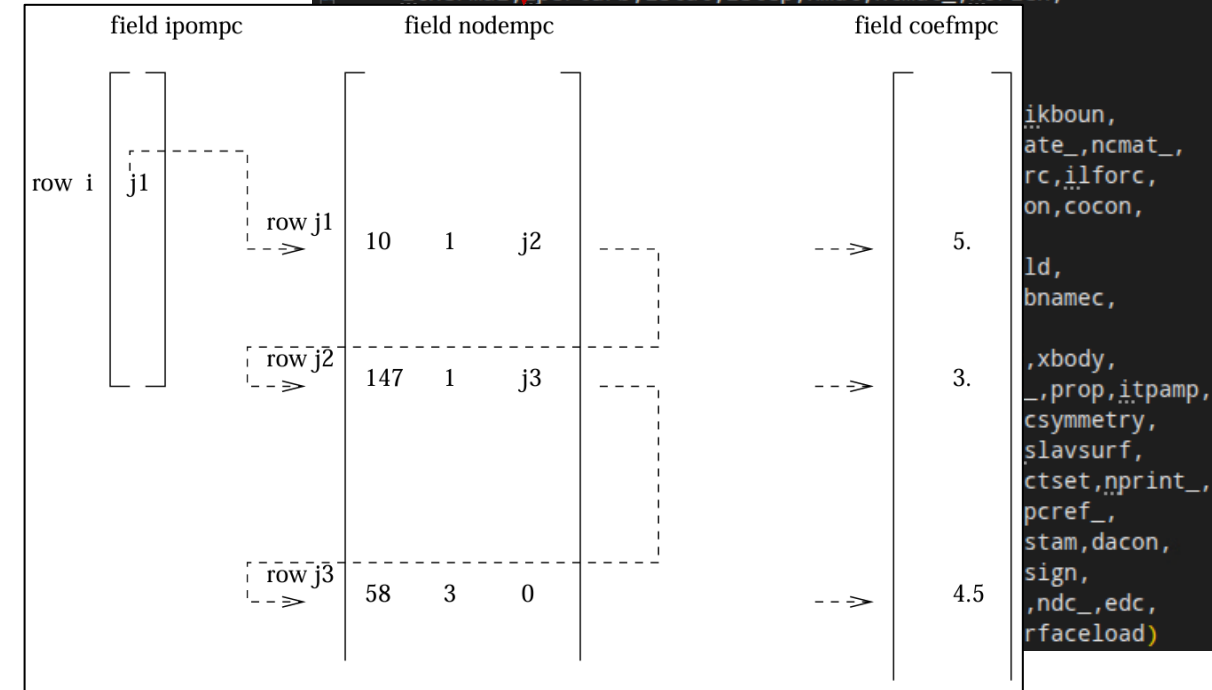
- Centralizing activity via GitHub and Discourse
- Enabling multi-contributor workflows
 - Contributing guidelines
 - Coding standards
 - Code reviews

Code Base Properties & Challenges

- Large, traditional, procedural mixed C/Fortran (fixed format) code (~250k SLOC)
 - C mainly as driver and memory management layer → requires careful manual management of dynamic memory
 - Virtually no abstractions / encapsulation → unwieldy function signatures (>100 arguments in extreme cases)
 - Implicit interfaces on Fortran side → modern Fortran features / checks unavailable
 - Interoperability through calling of raw symbols → some platform dependency (mitigated by extensive macro use)

Example: internal representation of MPCs

```
subroutine calinput(co,nk,kon,ipkon,lakon,nkon,
&...ne,noboun,ndirboun,xboun,nboun,
&...ipompc,nodempc,coefmpc nmpc,nmpc_,nodeforc,ndirforc,xforc,
&...nforc,
&...nforc_,nelemload,sideload,xload,nload,nload_,
&...nprint,prlab,prset,mpcfree,nboun_,
&...mei,set,istartset,iendset,ialset,nset,nalset,elcon,nelcon,
&...rhcon,
&...nrhcon,alcon,nalcon,alzero,t0,t1,
&...matname,ielmat,orname,orab,ielorien,amname,amta,namta,nam,
&...nmethod,iamforc,iamload,iamt1,
&...ithermal,iperturb,istat,istep,nmat,ntmat_,norien,
```



$$5 * u_1^{(10)} + 3 * u_1^{(147)} + 4.5 * u_3^{(58)} = 0$$

Code Base Properties & Challenges

- Support for legacy architectures / compilers / solvers → test automation for building & testing all combinations desirable
- Few external dependencies
 - Mostly handwritten algorithms also for standard functionality (e.g. geometric and sparse matrix operations)
 - Solvers and standard linear algebra packages (mainly SPOOLES, PARDISO, PaStiX, BLAS/LAPACK, ARPACK)
- Legacy output format (.frd) → large company-internal toolchain dependency but fewer open source tools (notably CalculiX GraphiX)
- Bare-bones Makefile → fragmentation of 3rd party build scripts / documentation across platforms
- Shared memory parallelization based on “parallel-region”-local POSIX threads

Example: $y = A \cdot x$ for real sparse symmetric matrices

```
subroutine op(x,y,ad,au,jq,irow,na,nb)
!
implicit none
!
integer irow(*),na,nb,j,l,i,jq(*)
!
real*8 y(*),x(*),au(*),ad(*)
!
! ..... diagonal terms
!
do i=na,nb
  y(i)=ad(i)*x(i)
enddo
!
! ..... off-diagonal terms
!
do j=na,nb
  do l=jq(j),jq(j+1)-1
    i=irow(l)
    y(i)=y(i)+au(l)*x(j)
    y(j)=y(j)+au(l)*x(i)
  enddo
enddo
!
return
end
```

Code Modernization

Goal 1: Reducing technical debt

- Replacing/updating legacy dependencies (e.g. hard-copied LAPACK routines, ARPACK, divergent forked PaStiX)
- Transitioning to OpenMP (see example)
- Introducing standardized C/Fortran interoperability using ISO_C_BINDING

Goal 2: Improving maintainability

- Gradually introducing abstractions and modularization → reducing code redundancies & function sizes
- Gradually building (in-code, parseable) developer documentation
- Introducing/enabling assistive tooling (e.g. linters, asan)

Goal 3: Improving utility

- Harmonization of build systems
- (Unified) interfacing to modern solver & linear algebra libraries
- Device offloading for special tasks

Replaceable by OpenMP loop
in applybounfem

```
// --- GLOBALS: all args duplicated as 1 pointers ---
int *nodeboun1, *nboun1, *nmpc1, *nfreestream1, *nsolidsurf1, ...;
double *xbounact1, *vold1, *v1, ...;

// --- C CALLER (compfluidfem.c) ---
void compfluidfem(...) {
    int num_cpus, ithread[num_cpus];
    pthread_t tid[num_cpus];
    // 1. Stash all args into globals
    nodeboun1 = nodeboun; nboun1 = nboun;
    // ... (25+ assignments)
    // 2. Launch threads
    for (i = 0; i < num_cpus; i++) {
        ithread[i] = i;
        pthread_create(&tid[i], NULL, applybounfemmt, &ithread[i]);
    }
    for (i = 0; i < num_cpus; i++)
        pthread_join(tid[i], NULL);
}

// --- C SCHEDULER ---
void *applybounfemmt(int *id) {
    // helper function for calculating the threads' loop bounds
    int nbouna = *id * ceil(*nboun1/num_cpus) + 1;
    int nbounb = min(nbouna + ceil(*nboun1/num_cpus), *nboun1);
    int nmpca = ...; // same pattern for each array
    int nmpcb = ...;
    int nfreestreama, nfreestreamb, ...;
    FORTRAN(applybounfem, (nodeboun1, ..., &nbouna, &nbounb,
                          &nmpca, &nmpcb,
                          &nfreestreama, &nfreestreamb,
                          &nsolidsurfa, &nsolidsurfb));
    // ↑
    // interop macro
}

// --- Fortran WORKER (applybounfem.f) ---
subroutine applybounfem(..., nbouna, nbounb, nmpca, nmpcb,
                      nfreestreama, nfreestreamb, nsolidsurfa, nsolidsurfb) {
    do j = nbouna, nbounb // SPCs
    do j = nmpca, nmpcb // MPCs (x3)
    do j = nfreestreama, nfreestreamb // freestream
    do j = nsolidsurfa, nsolidsurfb // solid surf
    }
}
```

Summary

→ Development and maintenance of CalculiX will continue 😊

Key points:

- Relatively large user base in comparison to number of developers
- Transitioning to a team-based development
- Improving accessibility for community contributions
- Maintaining stability and consistency

Other points on our current agenda:

- Implementation of the Riks solver (arc-length method)
- Improved linear solver interfaces
- User conference planned for Q4/2026 in Munich (will be announced via Discourse)

Development team

L. Bruder
C. Wölfle
G. Dhondt
M. Haßler
F.-J. Ertl
S. Teichtmeister
S. Erhard
T. Petzke