

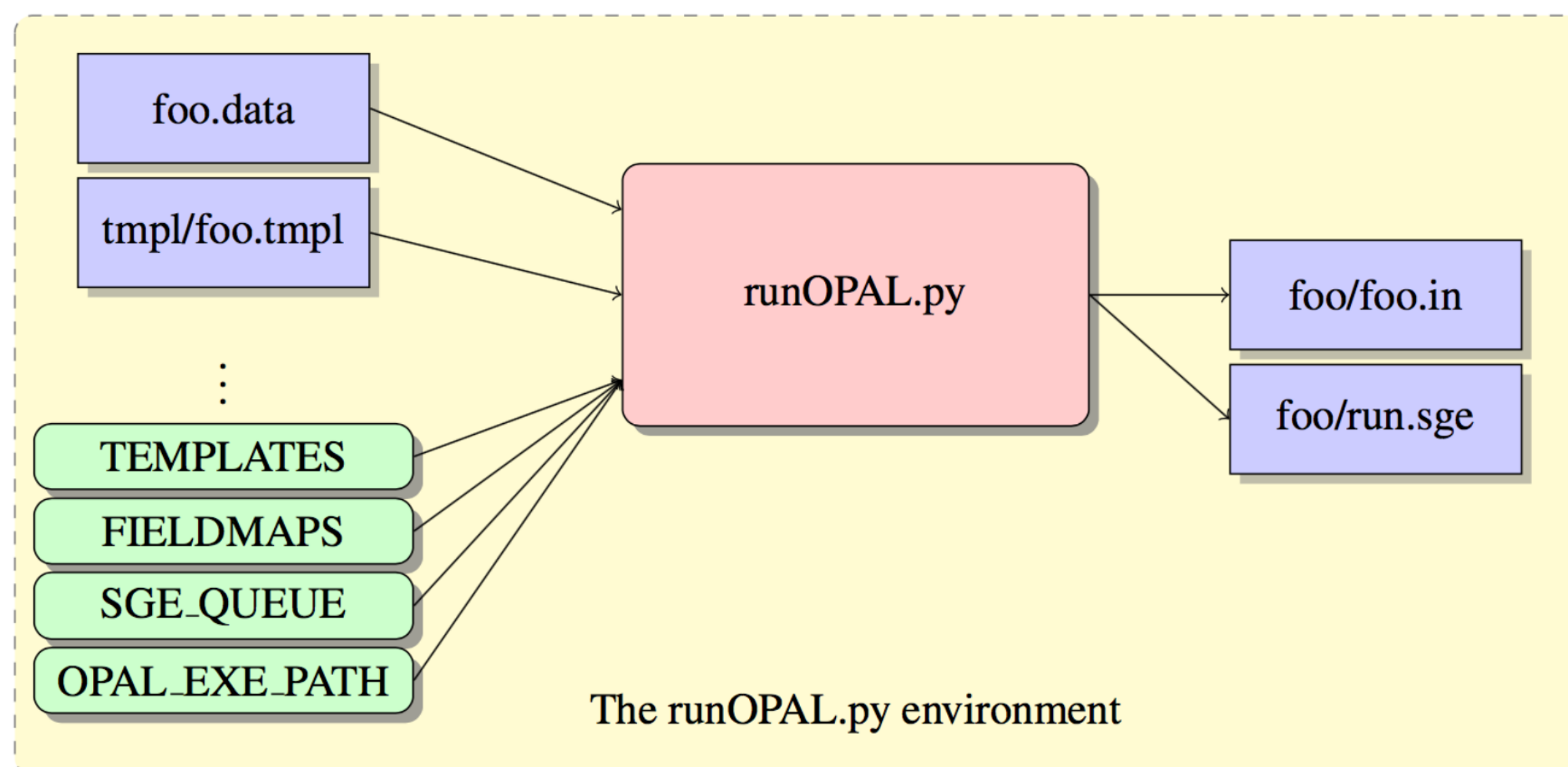
- pyOPALTools (<https://github.com/OPALX-project/pyOPALTools>)
- is modernised Python 3.x (5 months ago)

 amr	add file header to amr and surrogate files	6 years ago
 db	Modernize Python 3 docs and examples	5 months ago
 doc	Modernize Python 3 docs and examples	5 months ago
 jobhandler	add missing comma + add missing __init__.py	6 years ago
 opal	Modernize Python 3 docs and examples	5 months ago
 optPilot	remove argument from constructor and add it to t...	6 years ago
 pc	Modernize Python 3 docs and examples	5 months ago
 surrogate	Modernize Python 3 docs and examples	5 months ago
 tests	Modernize Python 3 docs and examples	5 months ago

 opal ○ what is left:: read .stat & .h5 + plot

If keep it who is taking care of ?

- runOPAL (<https://opalx-project.github.io/runOPAL>)
- is multidimensional scanning/submission tool



```
runOPAL.py --test EDES=0.050:0.250:0.050 POS1=0.4:0.5:0.01
```

If keep it who is taking care of ?

- Misc. Tools (<https://amas.pages.psi.ch/Tools/scripts/>)

- **Scripts**

Useful (and not so useful) scripts ...

Converting rfdata to T7

- Input: 1D field map (on-axis longitudinal electrical field)
- Output: T7-like 2D field data
- Calls: `interpolateFieldmap(fourCoefs, r, z, cavity_info)`
- Script: [interpolateFieldmapVarZnR.m](#)

Also: - Input: `rfdata` (array of Fourier coefficients)

- Output: longitudinal and radial electric field and transverse magnetic field at one position `(z, r)`

- Script: [interpolateFieldmap.m](#)

Example:

```
interpolateFieldmapVarZnR('INEG-ECAT.Ez', [0.0:0.01/1000:0.01], [0.0:0.001/100:0.001], [0.01, 0.01])
```



On this page

Converting rfdata to T7

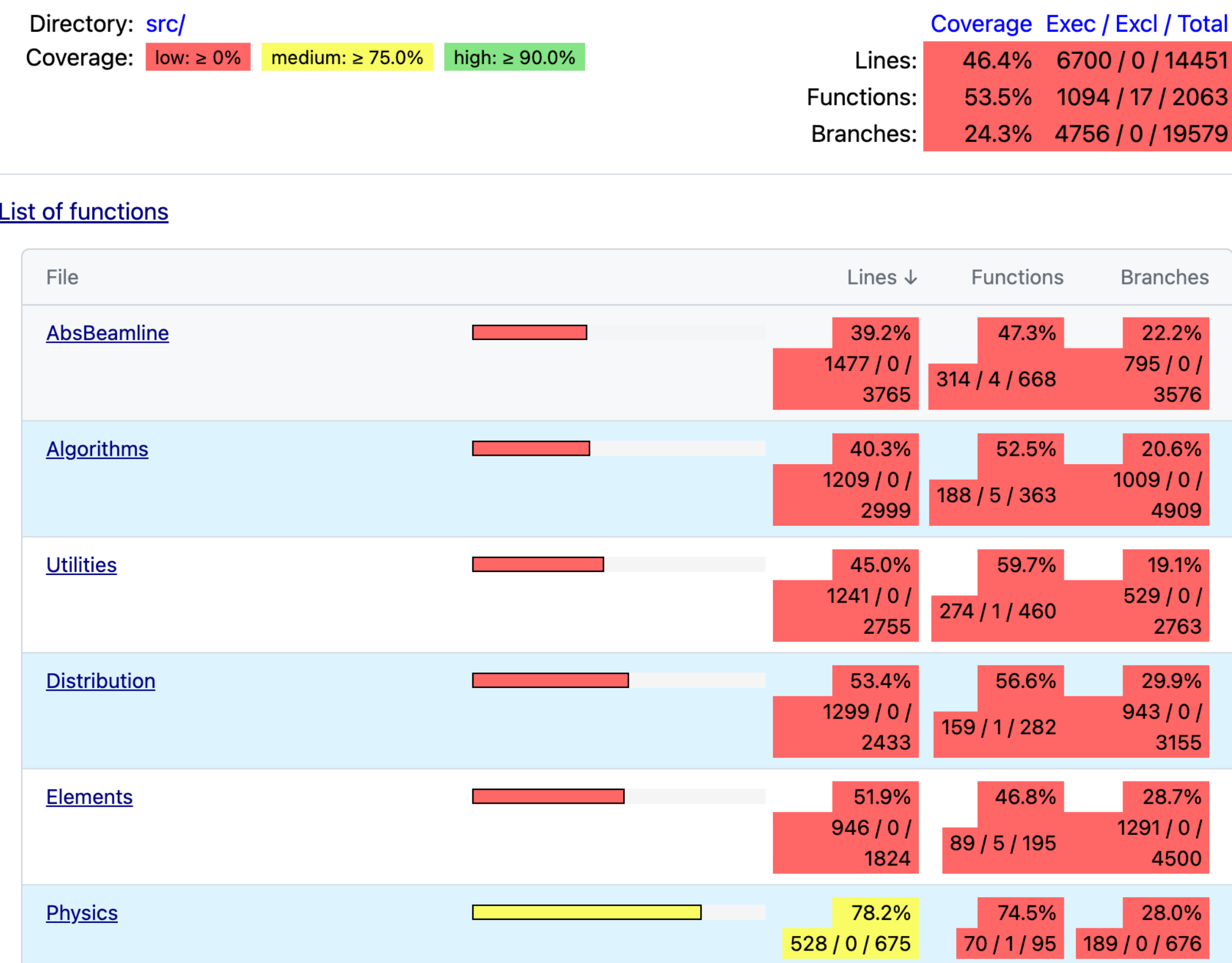
Converting non-uniform ASTRA field map to uniform OPAL field map

Converting HFSS to OPAL field maps

Comparing different reconstruction methods for field maps

Visualize T7 file of a standing wave cavity

- Code Coverage Report (CCR)
 - The current baseline is 38.2% line coverage which is not enough
 - Important and relevant parts of the OPALX are:
 - src/Physics/
 - src/Algorithms/
 - src/Distribution/
 - src/AbsBeamline/
 - src/Elements/
 - src/Utilities/
 - What is a sensible target?



Guidance from the outside

- Google's engineering guidance classifies 60% as “acceptable,” 75% as “commendable,” and 90% as “exemplary,” while noting diminishing returns above 90%. Google Testing Blog <https://testing.googleblog.com/2020/08/code-coverage-best-practices.html>
- For scientific software specifically, a peer-reviewed PLOS Computational Biology article suggests at least 60% as a rule of thumb, but stresses that the appropriate level is context-dependent and that branch coverage better reflects whether decision logic was exercised.
<https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1009481>
- Coverage alone is not a quality guarantee. A large Microsoft Research study of 100 open-source projects found no meaningful file-level relationship between coverage and post-release defects. <https://www.microsoft.com/en-us/research/publication/code-coverage-and-post-release-defects-a-large-scale-study-on-open-source-projects/>
- Applied to the measured six OPALX areas:
 - Current combined coverage: 46.4%.
 - First target: 60%, requiring approximately 1,971 additional covered production lines at the current denominator.
 - Longer-term target: 70–75%.
 - Physics is already 78.2%; prioritize meaningful branch, boundary-condition, conservation, and regression tests rather than merely increasing its percentage.
 - Low coverage in orchestration-heavy code such as trackers should also be addressed through integration, MPI, and end-to-end tests—not forced exclusively through gtest unit tests.
 - The practical CI policy should therefore be: never let existing coverage decrease, require high coverage on changed code, and gradually ratchet the aggregate baseline from 46.4% to 60%, then toward 70–75%.



Regression / Unit Tests



Testing data for this project can be found at: <https://github.com/OPALX-project/OPALX>

Experimental138 builds

View Timeline

		Configure		Build		Test				
Site	Build Name	Error	Warn	Error	Warn	Not Run	Fail	Pass	Start Time ▼	Labels
Eiger OpenMP	PR-524-cpu-zen2-openmp-Release-1-rank	0	0	0	0 ₋₄	0	0	86	44 minutes ago	(none)
Eiger OpenMP	PR-524-cpu-zen2-openmp-Release-4-ranks	0	0	0	0 ₋₄	0	0	86	44 minutes ago	(none)
Eiger OpenMP	523-add-checkpointrestart-capability-cpu-zen2-openmp-Debug-1-rank	0	0	0	0 ₋₄	0	0	86	44 minutes ago	(none)
Daint gh200 CUDA 13.1.1	523-add-checkpointrestart-capability-gpu-cuda-gh200-Debug-1-rank	0	0	0	0 ₋₄	0	0	86	49 minutes ago	(none)
Daint gh200 CUDA 13.1.1	523-add-checkpointrestart-capability-gpu-cuda-gh200-Release-1-rank	0	0	0	0 ₋₁₃	0	0 ₋₁	86	49 minutes ago	(none)
Daint gh200 CUDA 13.1.1	523-add-checkpointrestart-capability-gpu-cuda-gh200-Release-4-ranks	0	0	0	0 ₋₁₃	0	0	86 ₋₁	49 minutes ago	(none)
Daint gh200 CUDA 13.1.1	PR-524-gpu-cuda-gh200-Release-4-ranks	0	0	0	0	0	0	86	50 minutes ago	(none)
Daint gh200 CUDA 13.1.1										(none)
Daint gh200 CUDA 13.1.1										(none)
Eiger OpenMP										(none)

Published Runs

Newest row per branch and architecture

Branch	Architecture	Latest run	Build	Unit	Regression	Artifacts
pr-524 Add checkpoint and restart support	gpu-cuda-gh200	2026-08-25 22:37	external	85/86	250/250	Open overview
pr-524 Add checkpoint and restart support	cpu-zen2-openmp	2026-08-25 22:29	external	85/86	250/250	Open overview
pr-524 Add checkpoint and restart support	gpu-rocm-mi300	2026-08-25 20:08	external	86/86	250/250	Open overview