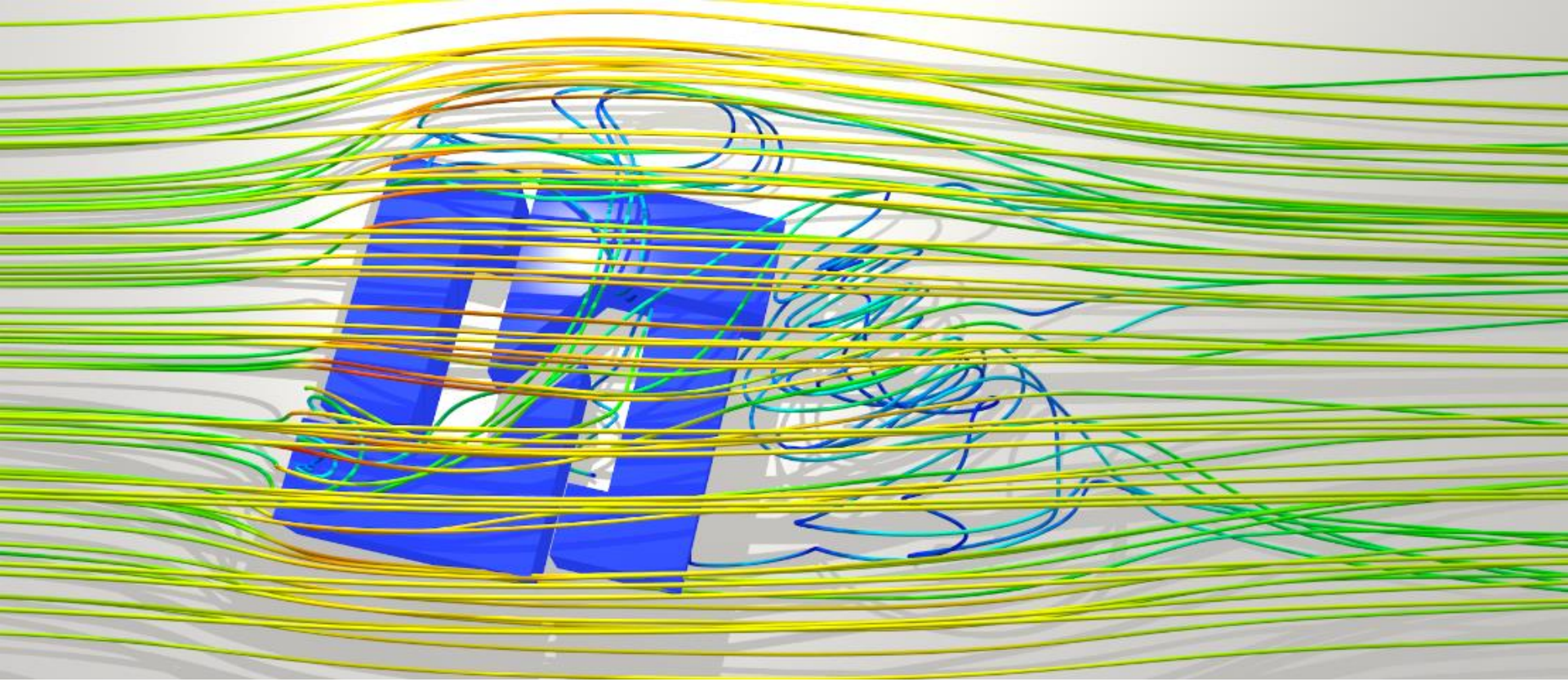
계산과학플랫폼(EDISON)에서 OpenFOAM 적용 사례

2019. 09. 26

넥스트폼 김태우

KiSTi www.kisti.re.kr 한국과학기술정보연구원 Korea Institute of Science and Technology Information

NEXTfoam open source CFD consulting



발표순서

1. 사이언스 앱 개발 방법
2. 워크플로우 적용 사례

사이언스 앱 개발 방법

- Data type 개발
  - OpenFOAM에서 입력이나 출력에 필요한 데이터 타입을 EDISON 포멧으로 개발
  - 각 solver 에 맞게 각종 데이터 타입을 생성하고 solver 에서 input이나 output 포트로 선택
  - 각 solver에서 생성한 결과들을 압축파일로 만든 후, 다음 solver의 입력 파일으로 받을 수 있는 openFOAM_link_result 타입 생성

데이터타입명	버전
blockMesh_boundary_input	V1.0.0
blockMesh_input	V1.0.0
openFoam_comp_results	V1.0.0
openFOAM_controDict_SDE	V1.0.0
openFoam_ControlDict	V1.0.0
openFoam_decomposeParDict	V1.0.0
openFoam_fvSchemeDict	V1.0.0
openFoam_fvSolutionDict	V1.0.0
openFoam_link_result	V1.0.0
openFoam_paraview_post	V1.0.0
openFoam_postResult	V1.0.0
openFoam_preResult	V1.0.0
openFOAM_SDE_input	V1.0.0
openFoam_transport	V1.0.0
openFoam_turbulence	V1.0.0

1 2

사이언스 앱 개발 방법

- Data type 개발
  - Character 값을 입력해야 하는 OpenFOAM 특성상, text_editor를 사용하여 주요 dictionary 파일 내용을 입력하도록 설정
  - integer값을 입력 받을 수 있는 경우, SDE 포멧으로 개발
    - blockMesh나 controlDict생성시 사용
    - endTime, deltaT, writeInterval 등

[text_editor]

[SDE]

사이언스 앱 개발 방법

- Solver 개발
  - Dictionary 파일 생성을 위한 converter 및 application 실행을 위한 solver 개발
  - 워크플로우 작성을 기준으로 solver 개발
  - 해석용 solver를 제외한 전/후처리 solver의 경우, 타 워크플로우에서 사용 가능하도록 개발

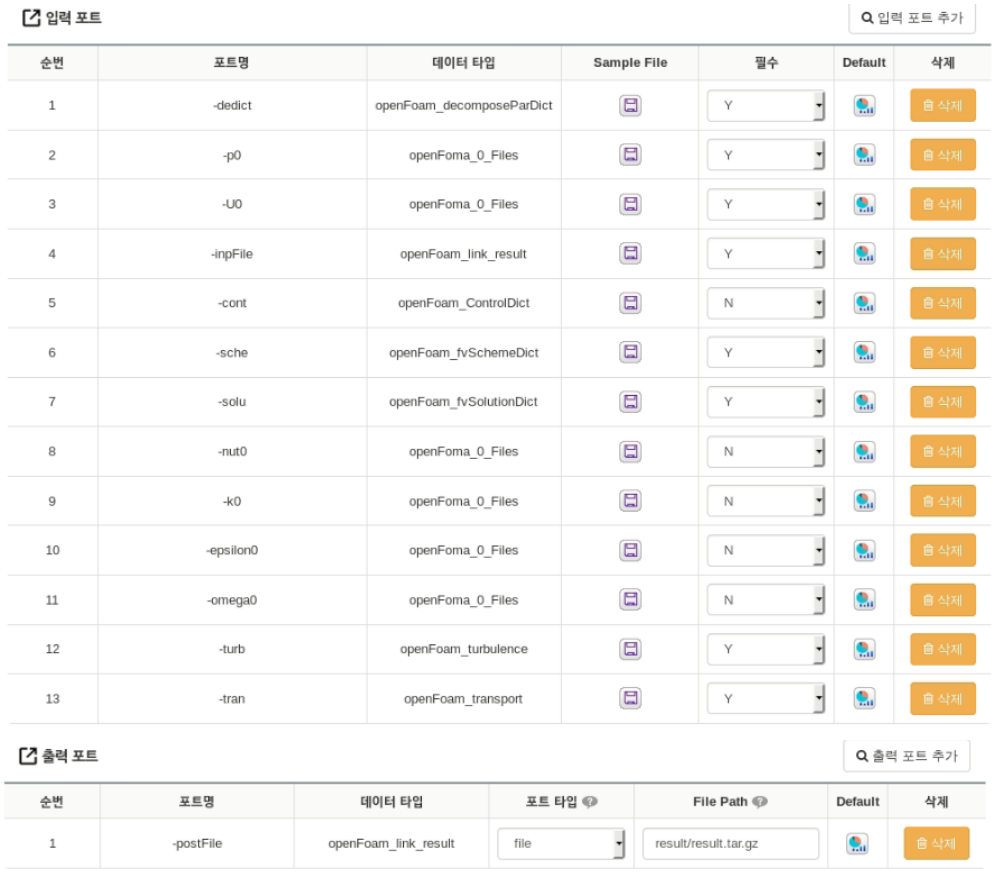
사이언스 앱

Clear | Filter

순번	타입	앱이름(앱제목)	상태	이름	기관명	최초등록일 / 최종수정일
7	Solver	foamToVTK_6.0.0 (Convert OpenFOAM results format to VTK)	공개	김태우	기타	2019-07-25 / 2019-07-25
6	Solver	reconstructPar_6.0.0 (Reconstruct decomposed mesh and fields)	공개	김태우	기타	2019-07-25 / 2019-07-25
5	Solver	renumberMesh_6.0.0 (Renumbering mesh band)	공개	김태우	기타	2019-07-25 / 2019-07-25
4	Solver	simpleFoam_6.0.0 (SIMPLE based steady-state imcompressible viscosity solver)	공개	김태우	기타	2019-07-24 / 2019-07-24
3	Solver	decomposePar_6.0.0 (Decompse domain for parallel computing)	공개	김태우	기타	2019-07-24 / 2019-08-05
2	Solver	blockMesh_6.0.0 (Hexahedral mesh generatior)	서비스요청	김태우	기타	2019-07-16 / 2019-07-24
1	Solver	icoFoam_6.0.0 (Transient solver for incompressible, laminar flow of Newtonian fluids)	공개	김태우	기타	2019-05-08 / 2019-07-09

사이언스 앱 개발 방법

- Solver 개발
  - 일부 solver의 항목의 경우, 반드시 필요한 입력 포트외에 해석 종류에 따라 필요한 입력 포트는 필수 flag를 N으로 하여 개발
  - 모든 해석자들은 result 폴더 내의 결과 폴더를 result.tar.gz로 생성하고 다음 단계에서 사용이 가능하도록 생성
  - foamToVTK와 같은 후처리 말단 solver의 경우, 출력 포트를 analyzer(paraview)가 사용하도록 개발


입력 포트

입력 포트 추가

순번	포트명	데이터 타입	Sample File	필수	Default	삭제
1	-dedict	openFoam_decomposeParDict		Y		삭제
2	-p0	openFoma_0_Files		Y		삭제
3	-U0	openFoma_0_Files		Y		삭제
4	-inpFile	openFoam_link_result		Y		삭제
5	-cont	openFoam_ControlDict		N		삭제
6	-sche	openFoam_fvSchemeDict		Y		삭제
7	-solu	openFoam_fvSolutionDict		Y		삭제
8	-nut0	openFoma_0_Files		N		삭제
9	-k0	openFoma_0_Files		N		삭제
10	-epsilon0	openFoma_0_Files		N		삭제
11	-omega0	openFoma_0_Files		N		삭제
12	-turb	openFoam_turbulence		Y		삭제
13	-tran	openFoam_transport		Y		삭제

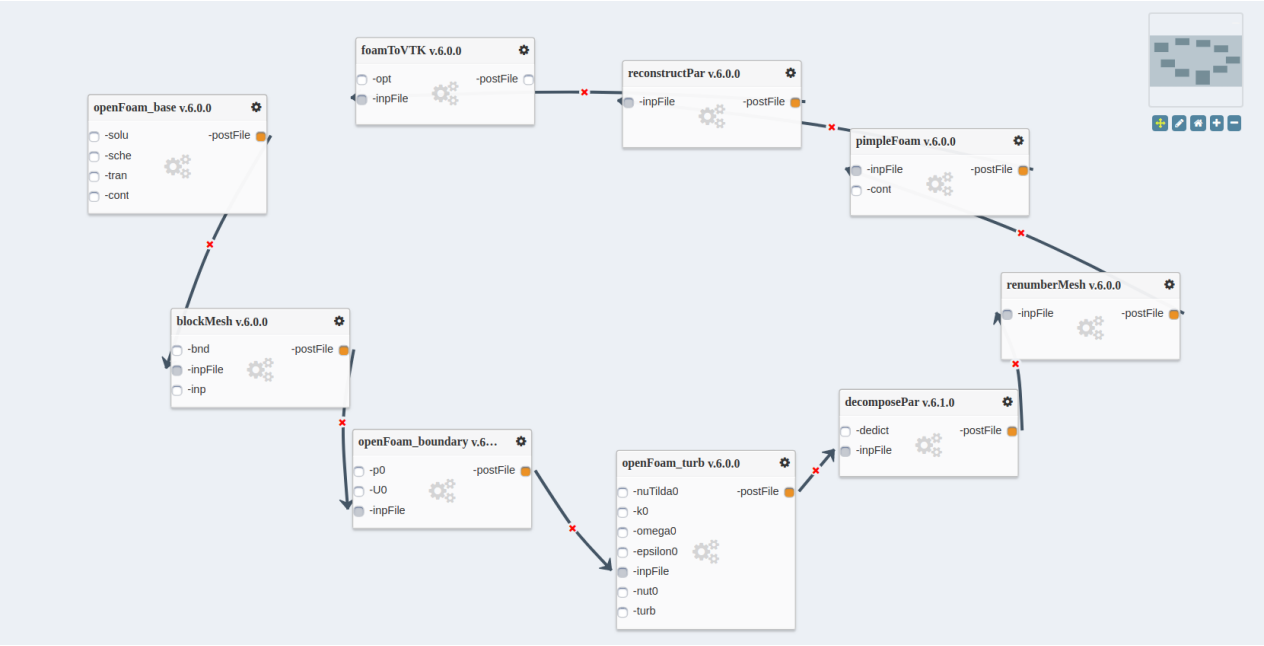
출력 포트

출력 포트 추가

순번	포트명	데이터 타입	포트 타입	File Path	Default	삭제
1	-postFile	openFoam_link_result	file	result/result.tar.gz		삭제

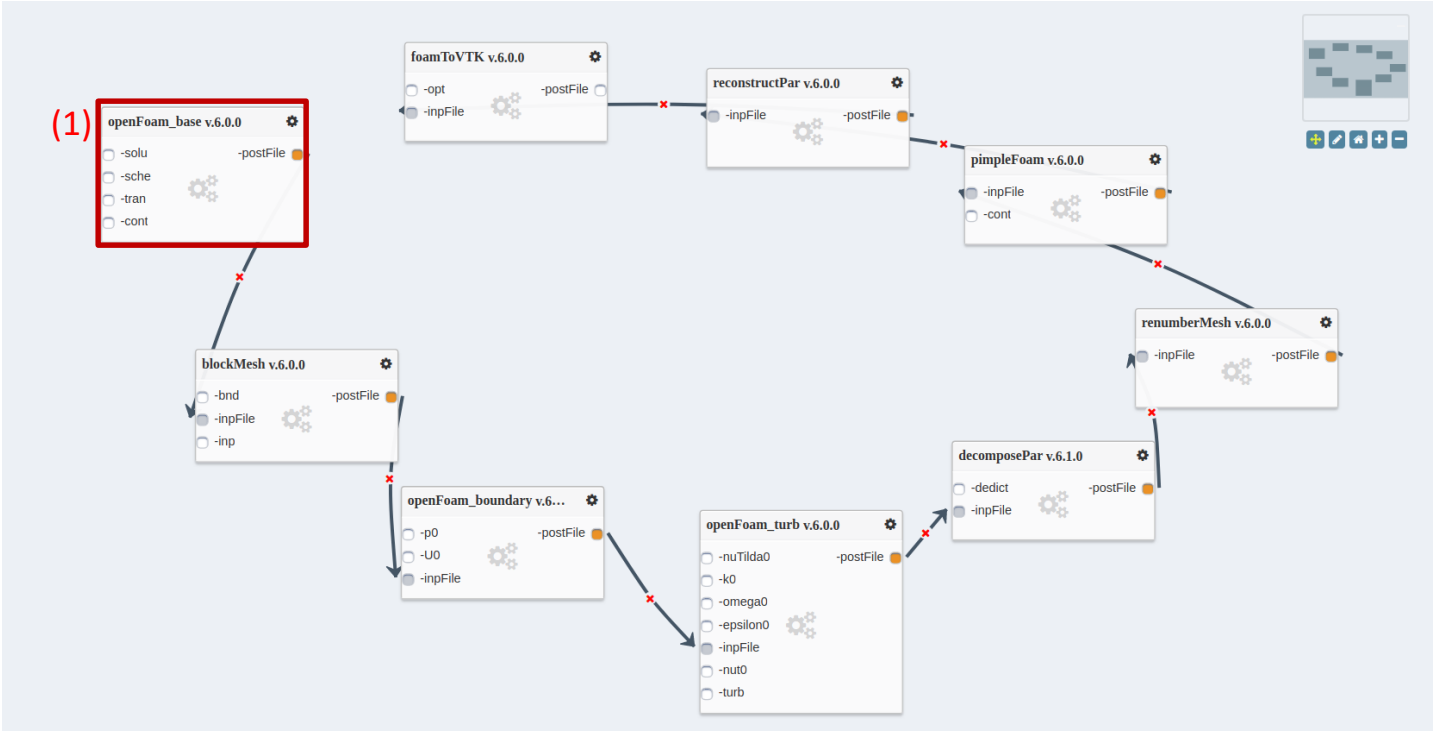
워크플로우 적용 사례

- 워크플로우 개발
  - 앞서 개발한 solver를 사용해서 워크플로우를 생성
  - solver 들의 결과 파일을 입력파일로 연결하여 전달
  - 도메인 분할에서 후처리까지 하나의 흐름으로 생성
  - 현재 simpleFoam(Steady), pimpleFoam(unsteady) 및 연전달해석 workflow 개발중



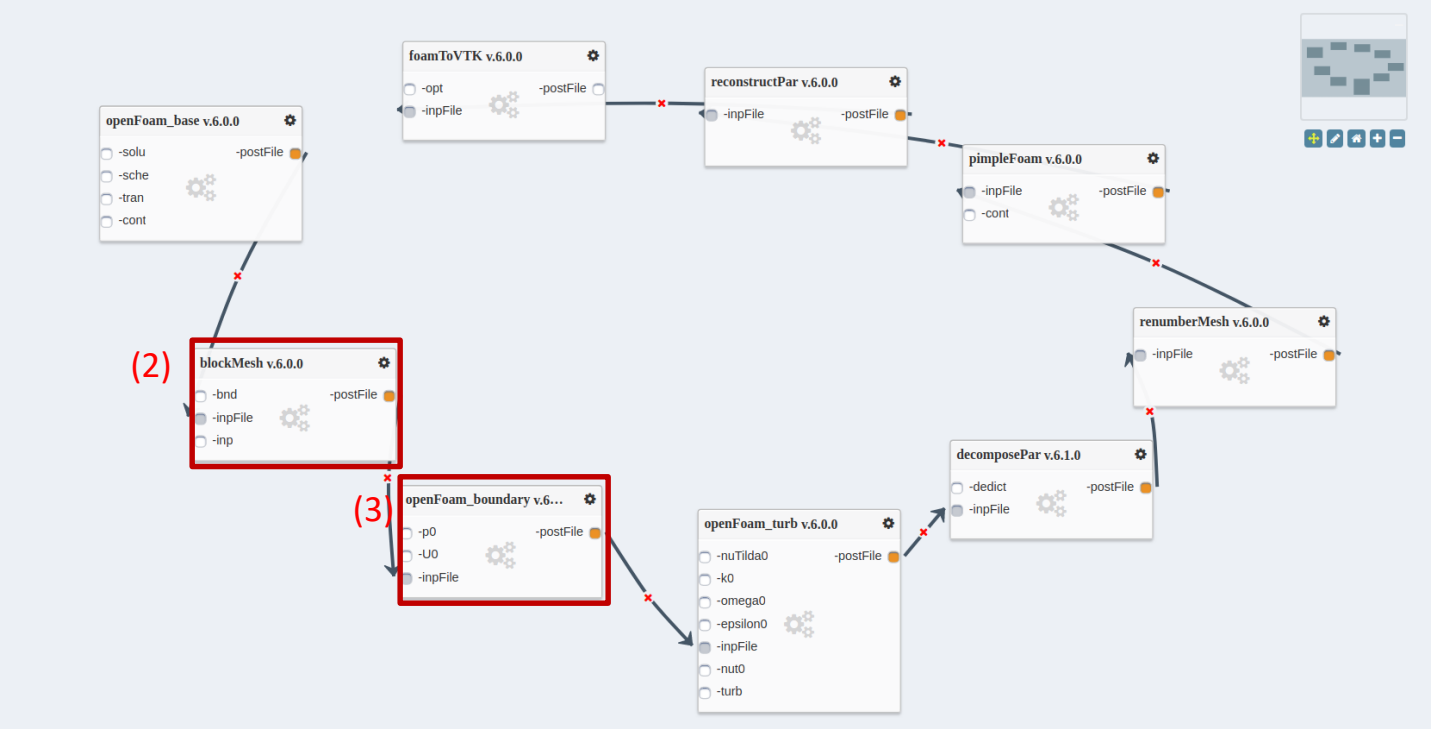
워크플로우 적용 사례

- 워크플로우 설명
  - (1)openFOAM_base
  - 가장 기본적인 control 및 transportProperty dictionary file을 생성
  - fvScheme과 fvSolution과 같은 해석 정확도에 관련된 dictionary 입력



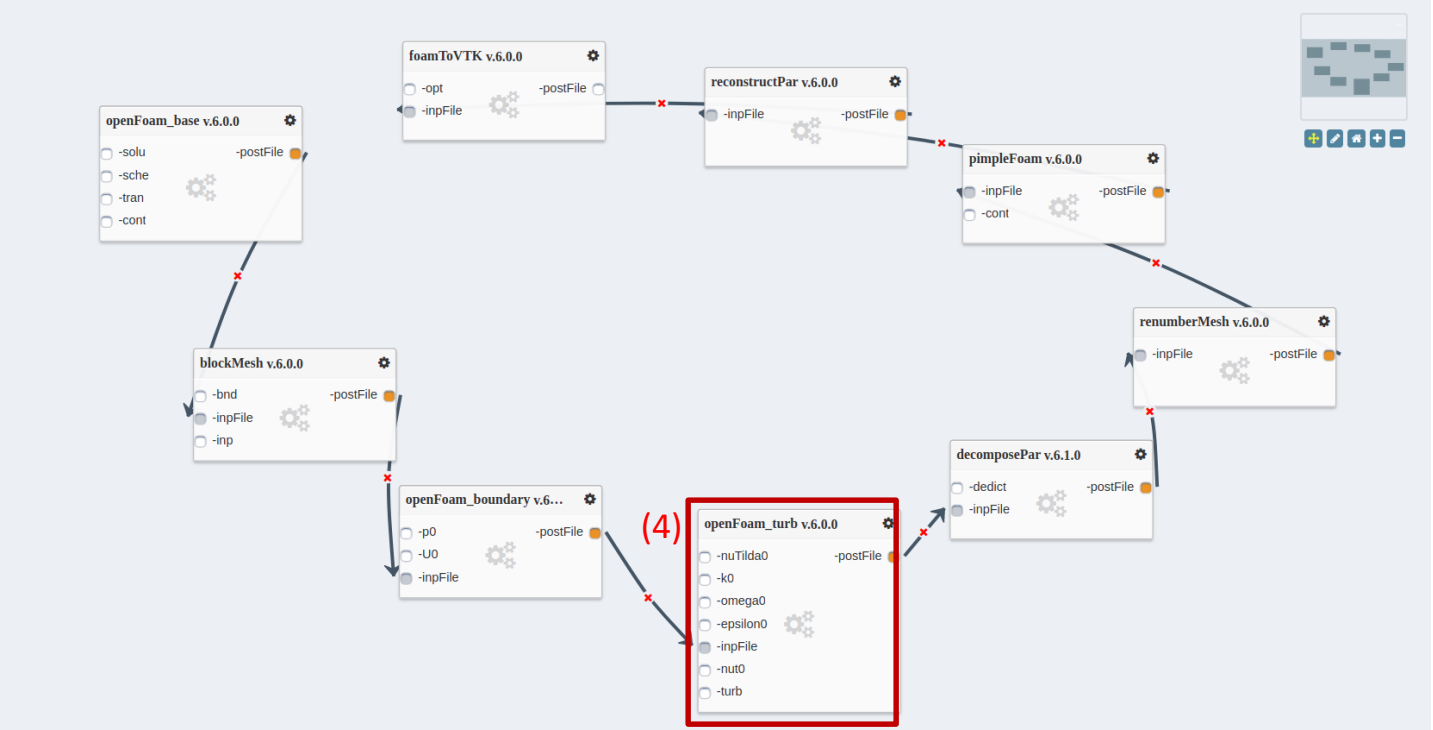
워크플로우 적용 사례

- 워크플로우 설명
  - (2)blockMesh : 앞에서 전달받은 파일과 blockMeshDict의 내용을 입력 받아 해석하는 격자 생성
  - (3)openFoam_boundary : 속도(U) 및 압력(p)에 대한 초기값 및 경계 조건 입력



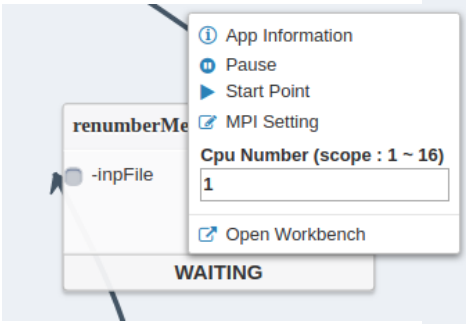
워크플로우 적용 사례

- 워크플로우 설명
  - (4)openFoam_turb : 난류 모델의 선택과 관련된 dictionary 파일과 난류 모델에 따른 변수들의 초기값 및 경계 조건 입력
  - 난류 모델에 따라 입력 받은 난류 변수들의 종류가 달라짐

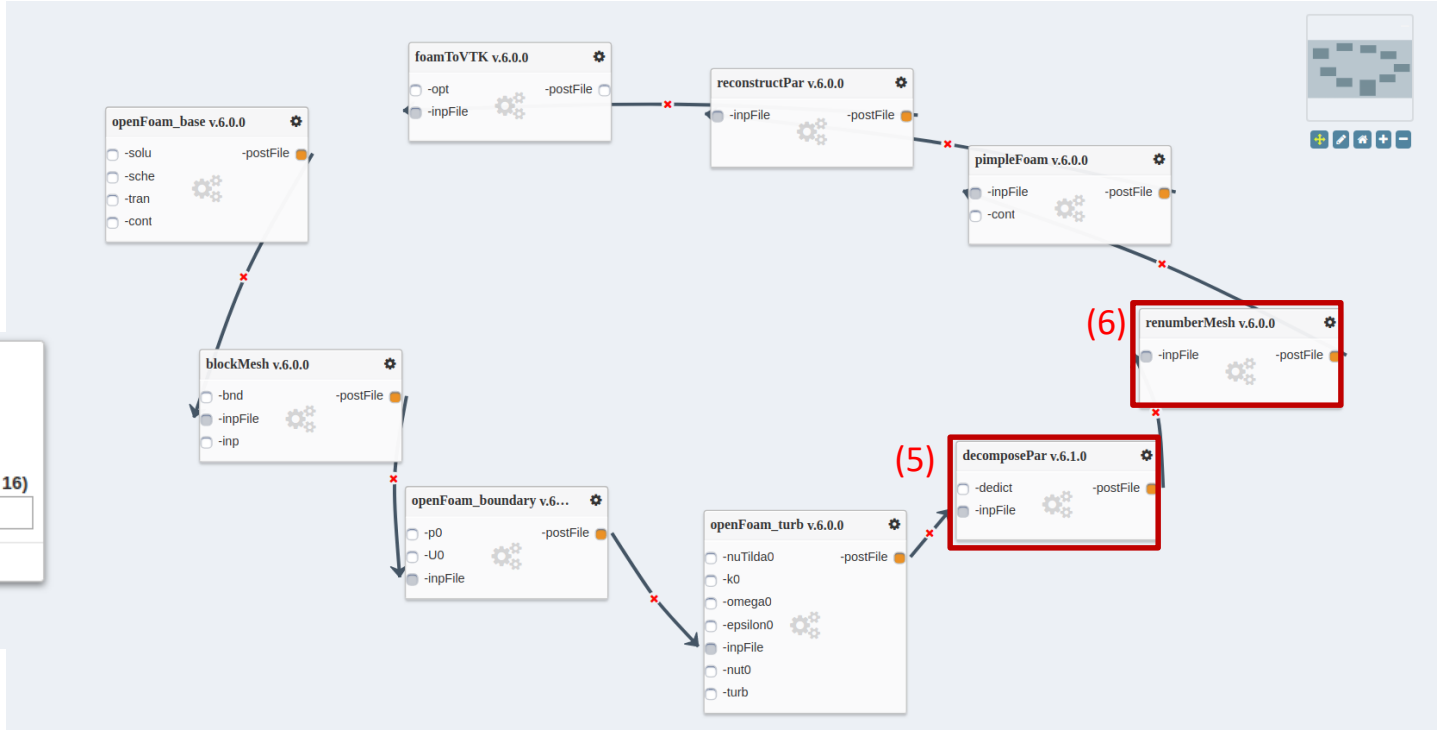


워크플로우 적용 사례

- 워크플로우 설명
  - (5)decomposePar : 병렬 계산을 위해 전달된 격자 및 변수를 프로세스 수대로 분할
  - (6)renumberMesh : 분할되거나 되지 않은 격자의 reordering을 수행
  - renumberMesh의 경우, 병렬 계산을 위해 CPU number를 설정하여야 함

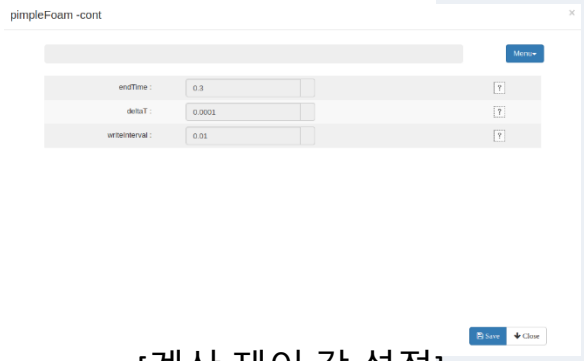


[병렬 계산 설정]

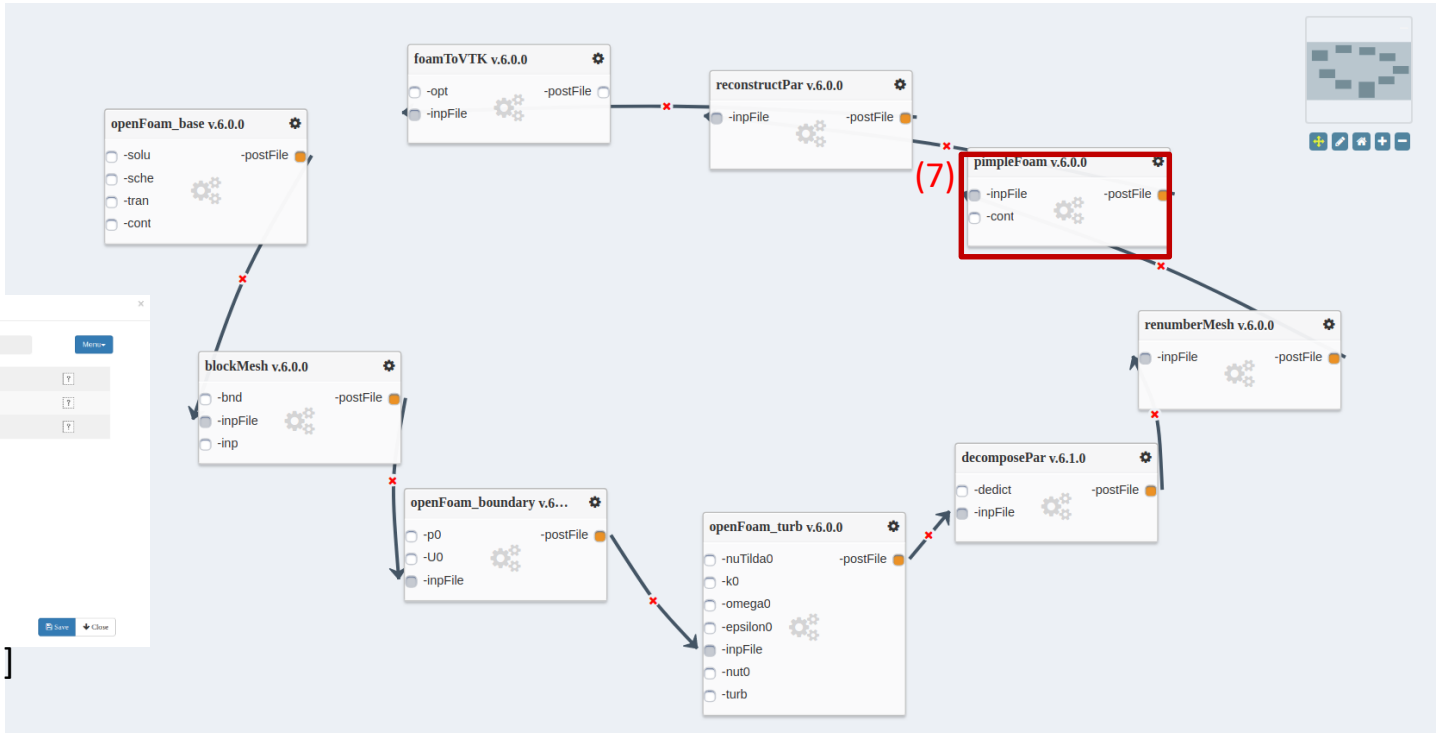


워크플로우 적용 사례

- 워크플로우 설명
  - (7)pimpleFoam : 실제 계산을 수행하는 solver를 구동
  - pimpleFoam의 경우, 병렬 계산을 위해 CPU number를 설정하여야 함
  - cont 포트를 통해 계산 종료 시간(endTime), 시간간격(deltaT), 저장간격(writeInterval) 설정

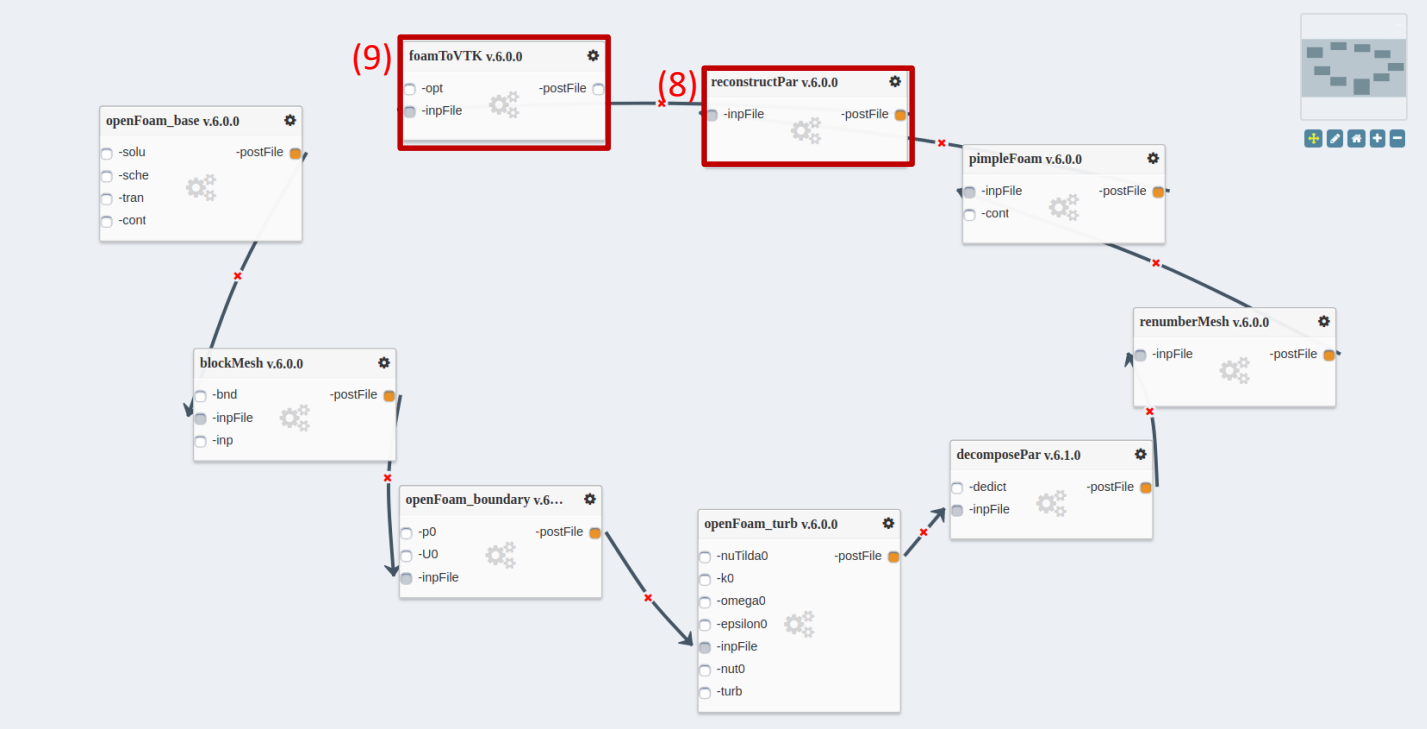


[계산 제어 값 설정]



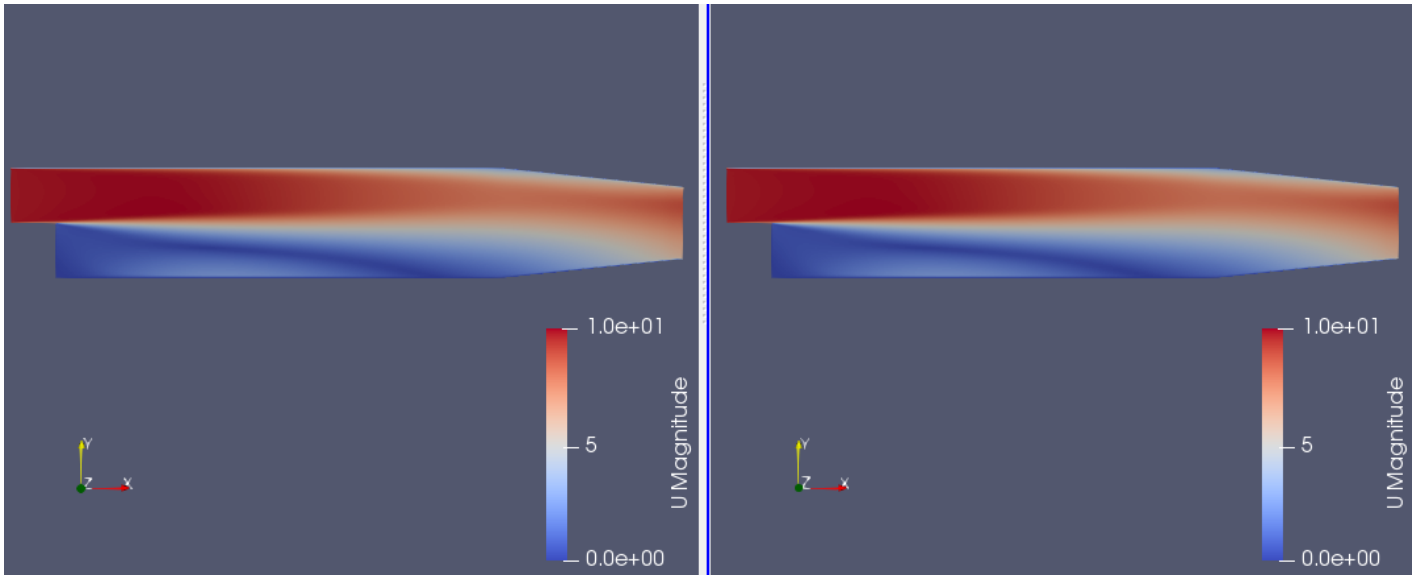
워크플로우 적용 사례

- 워크플로우 설명
  - (8)reconstructPar : 병렬 계산을 위해 분할된 도메인을 하나의 도메인으로 결합
  - (9)foamToVTK : 계산 결과를 VTK 포멧으로 변환하여 EDISON내의 post-processing을 통해 확인

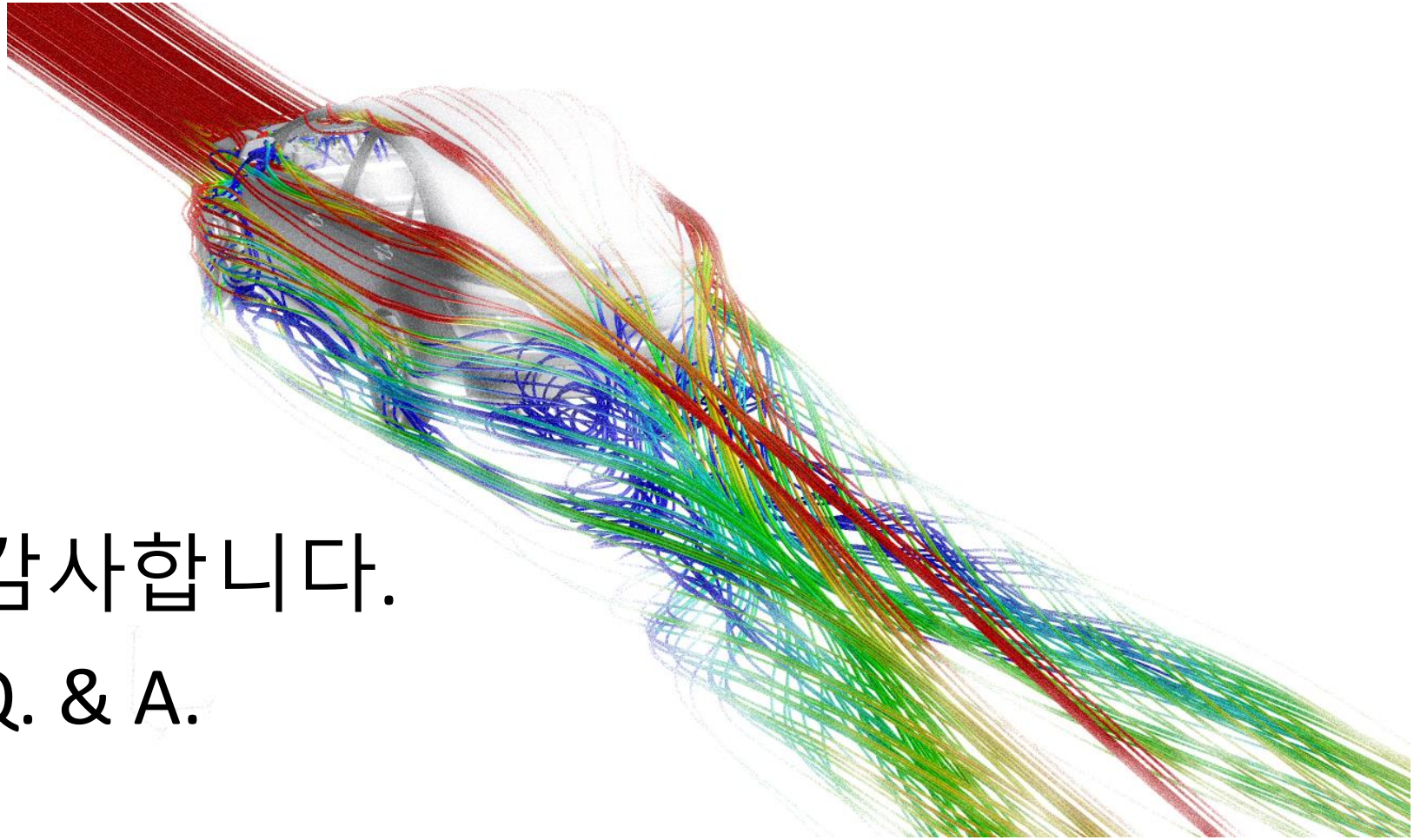


워크플로우 적용 사례

- 워크플로우 해석 결과
  - EDISON-Workflow에서 해석한 결과와 로컬 워크스테이션에서 구동된 결과는 동일
  - Native OpenFoam의 경우, TEXTUI 기반으로 작업을 수행해야하고 필요한 명령어를 하나씩 입력하거나, batch-shell 스크립터를 별도로 작성
  - EDISON-Workflow는 구성된 workflow를 사용하여 재계산시에 유리하며, 해석 절차 수립을 가시적으로 수행할 수 있어서 기술 이전에 효율적



[EDISON-Workflow] [Native OpenFOAM]



- 감사합니다.
- Q. & A.