# Task 2
# Contenu

**Objectifs :**

- Développer de nouveaux front-ends (application Linux ARMv8, source C++)
- Intégrer des back-ends
- Permettre à l'analyse d'interagir avec des logiciels tiers (bibliothèques)
- Gérer l'annotation du code pour exprimer l'impact de certains algorithmes sur la précision (par exemple, les algorithmes compensés).

**Programme de travail :**

- T[0-24] : Ajouter de nouveaux frontaux pour augmenter la classe des cas couverts (ISAs, OS, langages de programmation).
- T[12-36] : Développer une infrastructure pour interagir avec le code non-instrumenté et gérer l'annotation du code

# Task 2
# Deliverables

| | | | | | Statut | |
|---|---|---|---|---|---|---|
| [D2.1] | Front-end multiplateforme (analyse d'exécution pour Linux/Windows) | | @24 | @24 | OK. @30 | UPVD, UVSQ,, CEA List, **ANEO** |
| [D2.2] | Mémoire observationnelle pour propager des types de virgule flottante plus informatifs. | | @24 | | OK ? @30 | |
| [D2.3] | Mécanisme pour gérer l'annotation du code (@36) et interagir avec les bibliothèques externes (@24). | | @24-@36 | | KO ? | |

# Task 2 (1/2)
# New front-ends

- Verrou - EDF
  - Couverture par basic bloc dans le frontend verrou (l'utilisation de cette fonctionnalité est dans la tache 5)
  - Ajout de la racine carrée dans verrou
- Front-end C++ (preuve de concept) - CEA
  - ne fonctionne pas dans tous les cas, mais constitue une bonne base de travail.
  - Echoue lorsque les fonctions C free et malloc sont utilisées et lorsque certaines opérations à base d'union pour récupérer la mantisse et l'exposant sont utilisées.
  - Extension: rajout d'un identificateur unique d'instruction à chaque surcharge d'opérateur grâce à clang
  - Future extension: rajout d'un identificateur unique de type dans les opérations grâce à clang (pour la précision mixte)
- Unisim-VP – CEA
  - Preuve de concept de front-end armv8
- Verificarlo – UVSQ
  - Prototype Insane : LLVM + Shadow (C. Courbet + M. Jam), preuve de concept non maintenue.
  - Portage de Verificarlo sur ARM en cours [fonctionnel sauf pour les fonctionnalités d'interception de fonctions]
  - Implémentation de interflop_call [permettant une communication backend/frontend]
- Pene – ANEO
  - Support Windows Linux
  - Intégration des jeux d'instruction x86_64 jusque AVX512
  - Travaux sur backend vectorisé

# Task 2 (2/2)
# New mechanisms to collect information

- Verrou - EDF
  - Ajout de la racine carrée dans verrou
- Nouveau back-end "origins" - CEA
  - Compatible FLDLib pour inférer l'origine des erreurs et générer des pires cas.
- Fluctuat et FLDLib - CEA
  - Travail sur l'inférence d'erreur conservatrice à base de formes affines
  - Conception d'un domaine inférant une erreur relative "garantie" avec un intervalle de définition
  - Mécanismes de subdivision de Fluctuat en cours de transfert pour FLDLib
- Nouveau backend MCA entier – UVSQ
  - Inspiré de StochasticRounding.jl de M. Klöwer
- Autres backend verificarlo – CEA
  - Optimisation de performance [en particulier avec xoshiro]
  - Backends réentrants (applications parallèles / OpenMP)
- Nouveaux Backend-Pene - ANEO
  - Backend de test : couverture de l'instrumentation
  - Travaux en cours pour intégrer un backend unique dans PENE, VERROU, et Verificarlo

# Task 6
# Contenu

**Objectifs :**
- Valider Interflop sur des codes industriels existants
- Proposer des améliorations numériques sur ces codes

**Programme de travail :**
- Mises à jour trimestrielles sur le projet et réunion annuelle F2F avec les développeurs d'applications.
- T[12-36] : 6.1 Avec la Tâche1, tester sur toutes les applications la nouvelle fonctionnalité API
- T[24-48] : 6.2 Avec la Tâche2, tester sur toutes les applications le nouveau front, backend.
- T[0-48] : 6.3 Avec la Tâche3
  - (Localisation) : Code_aster, Telemac, Aghora, EuroPleXus
  - (Analyse composite) : proto-app de tous les codes qui nécessitent une analyse plus approfondie.
- T[0-48] : 6.4 Avec Task4 (mixte) : Yales2, Aghora, VamosLSA
- T[12-48] : 6.5 Avec Task5 (post-processing) : co-design post mortem et analyse en ligne.

# Task 6
# Deliverables

| | | | | | Statut | |
|---|---|---|---|---|---|---|
| [D6.1] | Pour chaque cas d'utilisation, rapports d'expérimentation avec la liste des améliorations (numériques et de performance) et le retour des utilisateurs finaux. | | @45 | | A formaliser | **INTEL, ANEO** |
| [D6.2] | Après chaque période d'un an, produire une feuille de route avec les prochains défis identifiés avec les utilisateurs finaux. | | @12 | | A formaliser | |

# Task 6
# Application Cases

- EDF
  - Utilisation de verrou, verrou_plot_stat, verrou_dd_line et post_verrou_dd (avec couverture par BB) sur code_aster.
    - Les outils fonctionnent après adaptation au cas multi-processus
    - Les localisations restent difficiles  (elles doivent être améliorées et il est indispensables de lutter contre les faux positifs cf. random_det and co)
    - Perfs de verrou_dd_line probablement améliorables
  - Utilisation de verrou et verrou_dd_line sur Telemac (cas barrage de malpasset)
    - Les outils fonctionnent
    - Sur un cas test random_det n'a pas le même comportement que random
    - L'analyse des localisations restent difficiles
    - La visualisation sur maillage des estimateurs d'erreur reste délicate
  - REX  commun :
    - Les correctifs par algo compensé sont complexes à intégrer dans un code industriel.
    - Je n'ai pas encore réussi à passer le relais dans les équipes métier sur l'utilisation de verrou.
- CEA
  - Outil propriétaire Fluctuat utilisable suite à sa qualification interne Airbus (février 2023)
    - La qualification v2 devrait utiliser FLDLib
    - vérifie sur les tests de qualification que FLDLib et Fluctuat ont une intersection non vide dans les résultats (soundness)
    - utilise la génération de pires cas de FLDLib/origins pour montrer que les résultats de Fluctuat sont précis (precision)
  - Utilisation de Fluctuat au sein d'Airbus pour démontrer que des composants ont une précision garantie.
- UVSQ
  - Utilisation de verificarlo au sein du Centre d'Excellence TREX (chimie quantique)