Common Geometry Primitives library

WP3

Mihaela Gheata, CERN EP/SFT

for the USolids/VecGeom team

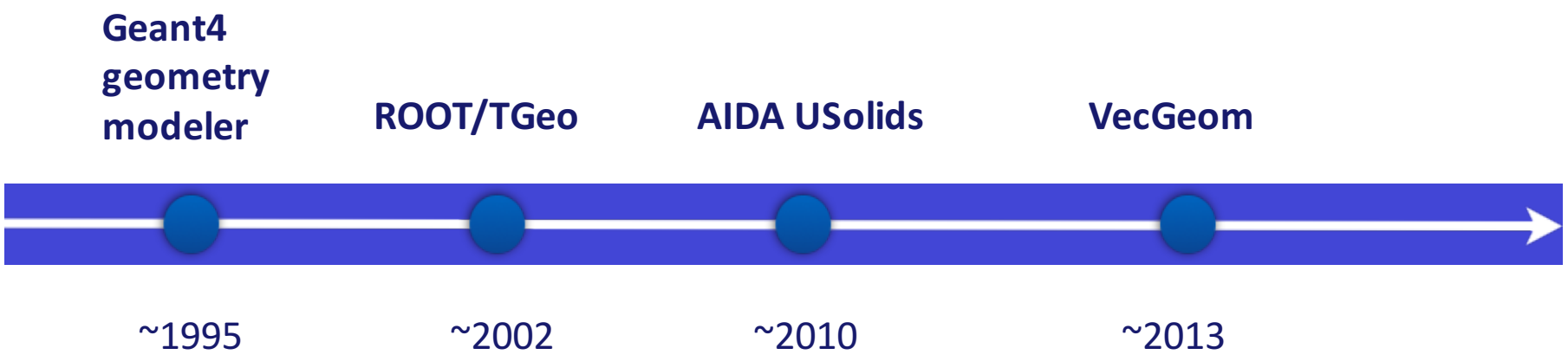
This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 654168.

Content

- Introduction
- Status
  - Solids implementation: completed and still missing
  - Interface restructuring within VecGeom
  - Migration to use VecCore
  - Recent and ongoing development on shapes
  - Interfaces with Geant4 and ROOT
  - Validation
- Directions and work plan

Motivations

- USolids (Unified Solids) library: started as AIDA project
  - Aiming to develop a new library of geometrical primitives to unify and improve algorithms existing in Geant4 and ROOT
- VecGeom: started as a project for developing a vectorised geometry modeler (as main requirement of GeantV)
  - Vectorization of algorithms, boost further performance and port to parallel architectures
- Geometry primitives code development as long-term evolution of USolids
  - VecGeom developed as independent library
  - Now incorporating USolids implementation and sharing same interfaces
  - Aiming to extend and totally replace USolids
  - Activity part of AIDA-2020/WP3



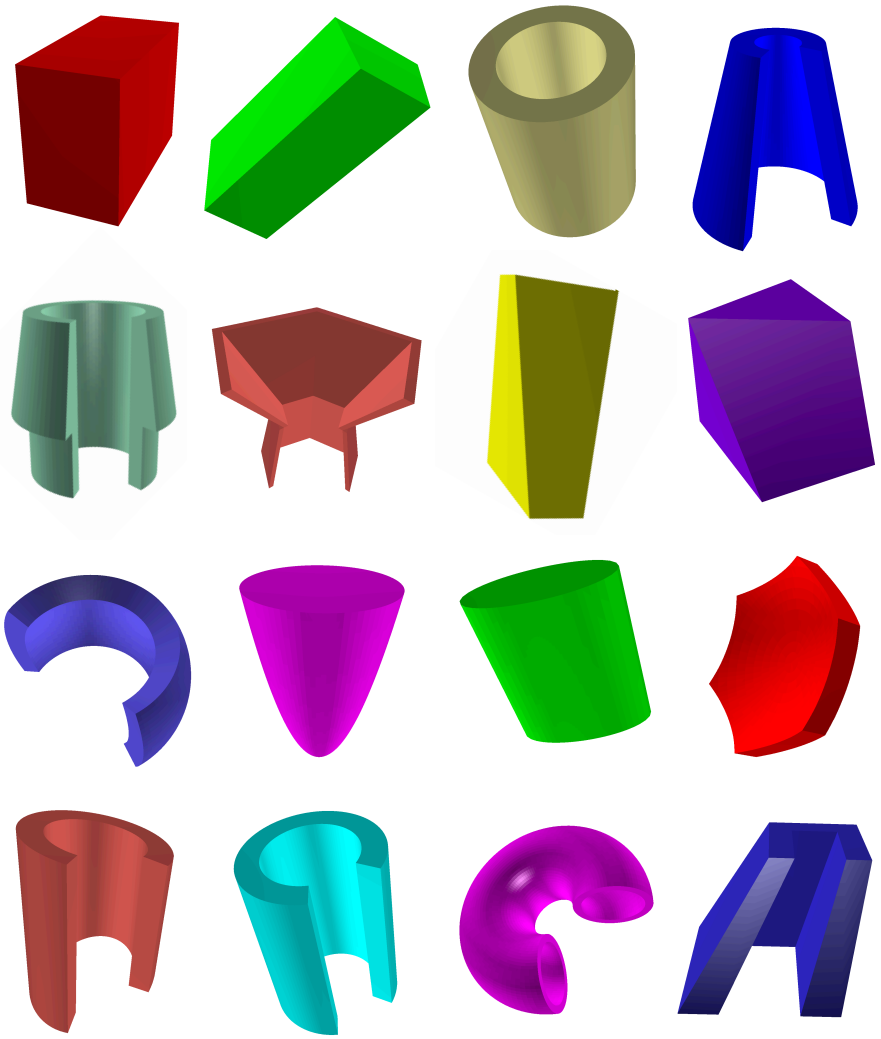
Shapes: implementation status

- Both in USolids and VecGeom:
  - Box, Orb, Trapezoid (Trap), Simple Trapezoid (Trd), Sphere (+ sphere section), Tube (+ cylindrical section) , Cone (+ conical section), Generic Trapezoid (Arb8), Polycone, Polyhedron
- Only in VecGeom:
  - Paraboloid, Parallelepiped (Para), Hyperboloid, Ellipsoid, Torus (+ torus section), Scaled Solid, Boolean (addition, subtraction, intersection), Cut Tube, Simple Extruded Solid (SExtru)
- Only in Usolids:
  - Tetrahedron (Tet -> GenericTrapezoid), Multi-Union, Tessellated Solid, Generic Polycone
  - Extruded solid

    expressed as specialization of tessellated-solid
- Missing:
  - Elliptical Cone, Elliptical Tube

    can be composed through scaling
  - Half-spaces/planes
  - Twisted shapes (box, trap, tube)

    complex and infrequent use



VecGeom interfaces restructuring

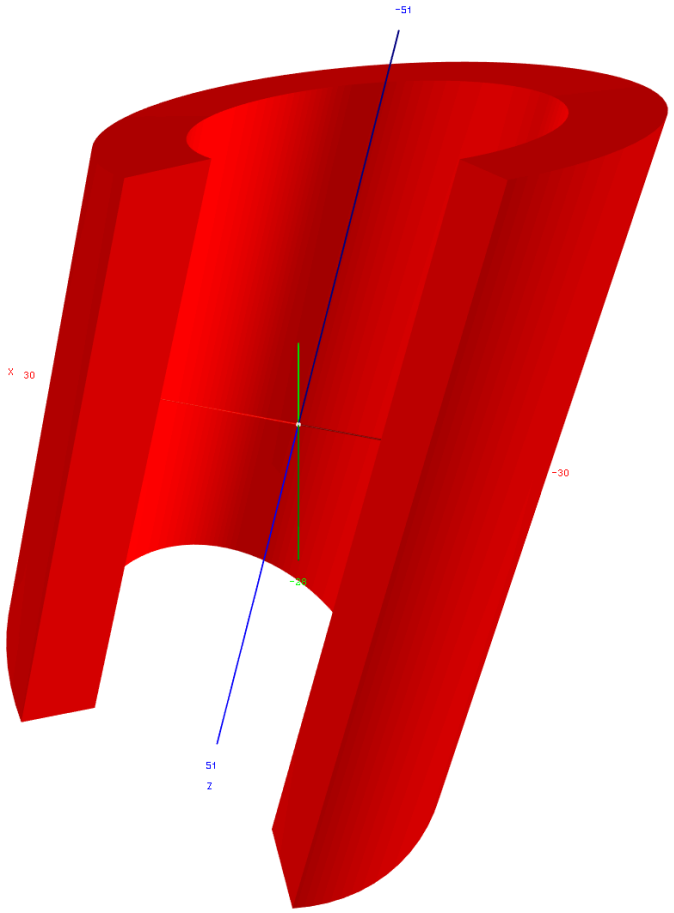
- The VecGeom main interfaces for navigation were attached only to placed volumes
  - UnplacedVolume ⇔ G4VSolid/TGeoShape; PlacedVolume = Unplaced + positioning
  - Not possible to call this functionality for unplaced volumes
  - Not possible to call specific static implementations without specifying the backend
    - template <typename Backend, …> BoxImplementation::Contains()
- Interfaces were restructured
  - Implementations are templated on types (vector/scalar) rather than on specific backends
  - Types are unified by a common abstraction (VecCore)
  - Possible now to call navigation from both placed/unplaced volumes
  - Possible to call specific implementation in a backend-independent manner
- Migration non-trivial for solids
  - Vectorized calls migrated to the new library (VecCore) - done
  - Code for solids being reshaped to follow new interfaces and types – in progress

VecGeom migration to VecCore

- **VecCore** is an open source SIMD abstraction library, initially developed within VecGeom.
  - It provides an architecture independent API for expressing vector operations on data.
- The VecCore library contains some implementations for the SIMD **backends** to access **vectorization libraries** (such as Vc or UME::SIMD)
  - "Backend" is a structure grouping vector SIMD types together for using them in generic algorithms
  - Backends have to support most basic operations: *construction, indexing, load/store, arithmetic operations, masking, math functions*
  - VecCore can be extended with new backends satisfying these requirements.
- VecCore is now externalized and available in GitHub
  - https://gitlab.cern.ch/VecGeom/VecCore
  - Usable as foundation library, adopted as well by ROOT
- VecGeom is fully migrated to use VecCore
  - Allowing easier specialization (types), cleaner interfaces formalized on VecCore types, code independent on specific backends/architectures

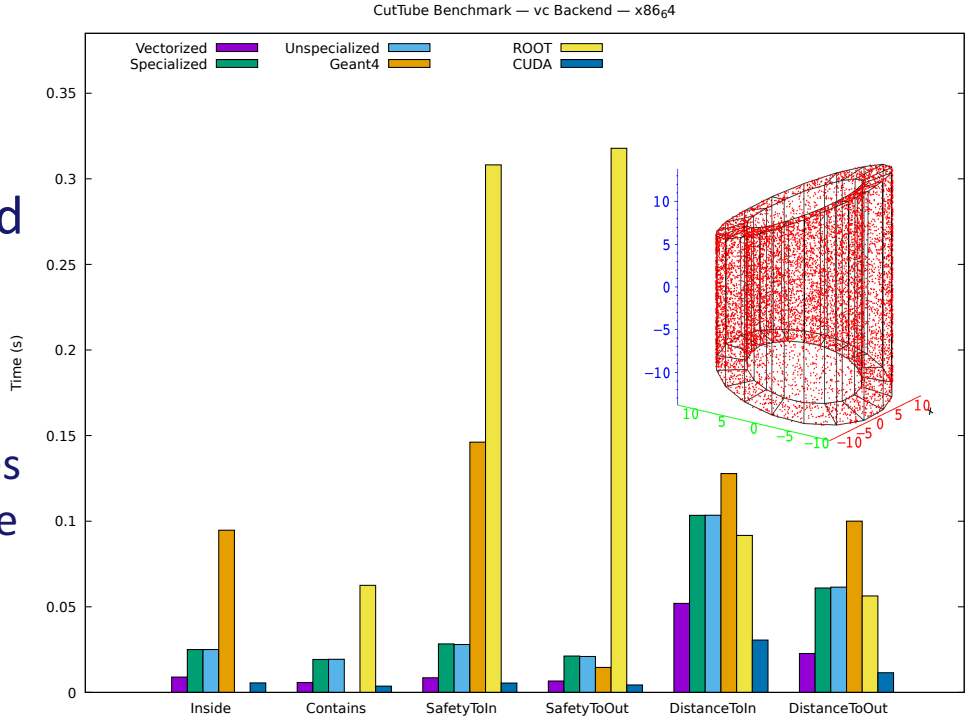
New solid: cut tube

- A tube cut by two planes
  - Planes cutting Z axis at +/- Z, defined by normal vectors oriented outwards the solid
- Available both in ROOT and Geant4 geometry packages
- Existing in ALICE and new CMS geometry
- Implemented using as primitives the tube and plane implementations
  - Plane implementation to be reused for half-space primitive, used for defining Boolean solids in ROOT



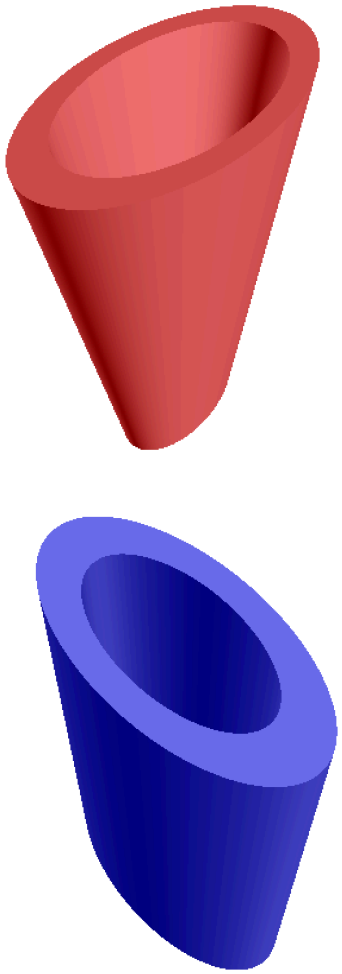
Cut Tube performance

- Overall scalar performance comparable to ROOT/Geant4
  - Much better for Inside, Contains and SafetyToIn
- Capacity and surface area computed analytically
  - Just estimates in Geant4 and ROOT
- SIMD and CUDA support
  - Implemented using VecCore interfaces
  - Vectorization gain for AVX in the range 2x-3x
- Visualizer, shape tester and benchmarker provided for 4 different topologies:
  - with/without Rmin
  - with/without phi cut



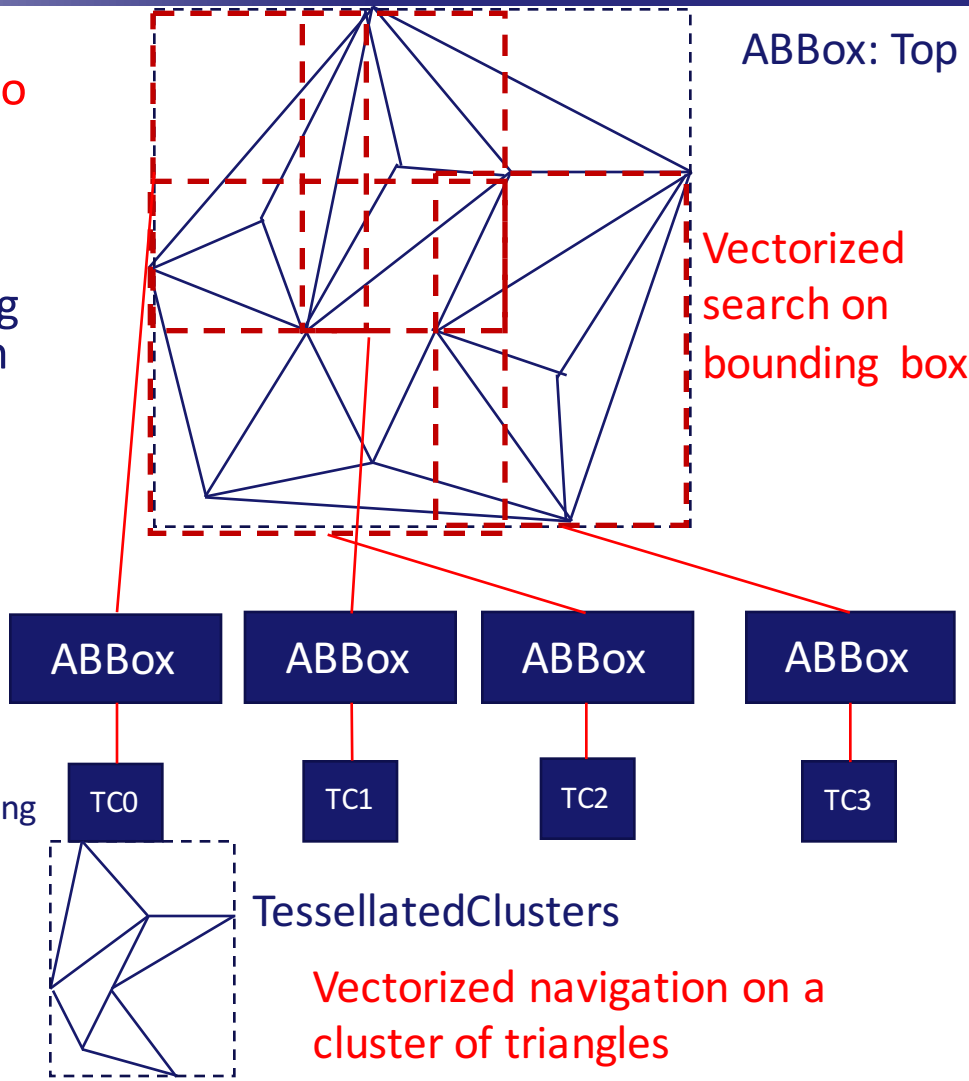
Scaled shapes

- An arbitrary scale transformation applied on any existing solid
  - Allows representing some missing solids
    - Cone -> Elliptical cone
    - Tube -> Elliptical tube
- Implementation inspired from ROOT
  - Calling the implementation of the solid to be scaled + scale transformations
- Performance = performance of solid to be scaled + small overhead



New solid: Tessellated Solid

- Implementation in progress, aiming to maximize vectorization
  - Starting from scalar implementation available in Geant4 and USolids
- Idea: group facets into clusters having the size equal to the vector length on the machine
  - Each cluster should contain a single facet type (triangle) to allow vectorization
- Two vectorization levels:
  - Cluster also bounding boxes of each triangle group to navigate hierarchically
    - Vectorized bounding box navigator existing already (used for volume daughters)
  - When reaching a cluster of triangles, vectorize on triangles



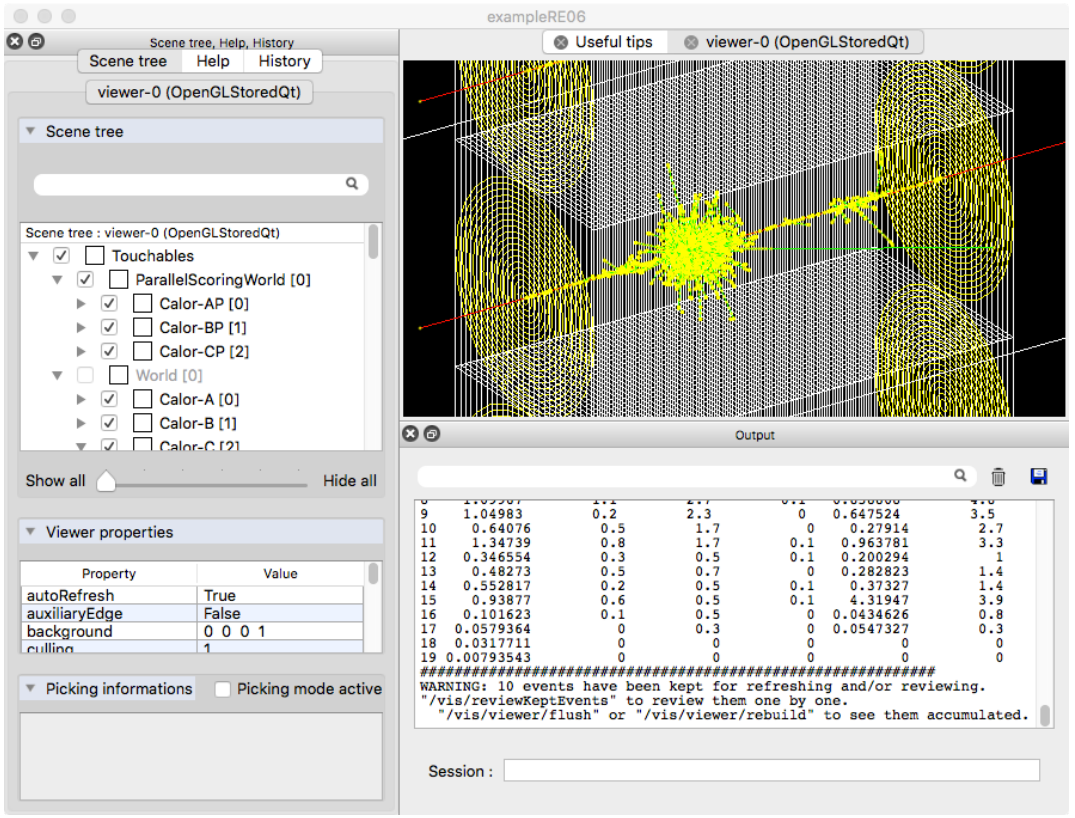
Status of testing validation and migration of shapes

Shape	Ray tracing precision response (*)	Stress tests (Shape Tester)
Box (o)	OK	OK
Tube (o)	OK	OK
Trapezoid (o)	OK	OK
Generic Trapezoid (Arb8) (o)	To be verified	OK
Simple Trapezoid (Trd) (o)	OK	OK
Orb (o)	OK	OK
Sphere (x)	Failures	OK
Cone(x)	OK	OK
Parallelepiped (Para) (o)	To be verified	OK
Polycone (x)	Failures	OK
Polyhedron (o)	OK	OK
Cut Tube (o)	To be verified	OK
Simple Extruded (Sextru) (o)	OK	OK
Torus (x)	OK	Failures

(*) Test on realistic detector geometries (o) Migrated to new framework interfaces (x) Being worked out now…

VecGeom solids used through Geant4

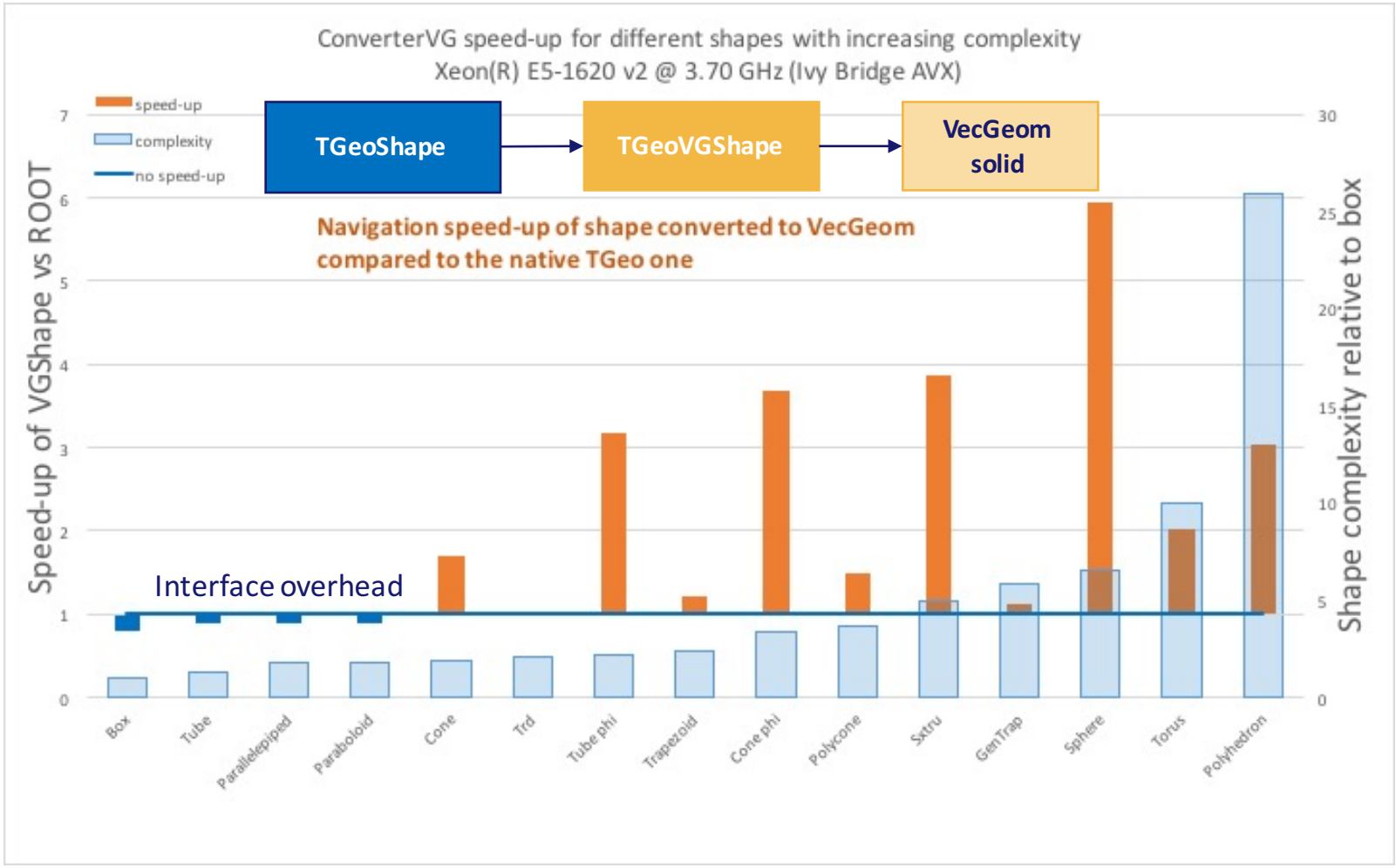
- VecGeom shapes can be used in Geant4 with either release 10.3 (latest) or 10.2
  - Passing majority of tests in the Geant4 testing suite
  - Part of nightly builds in Geant4
    - VecGeom shapes from Git master
    - Tested on CentOS/gcc-5.3
  - Exercised shapes:
    - Box, Tube, Cone, Orb, Sphere, Trap, Trd, Torus, Polycone, Polyhedron, Tet, GenericPolycone, GenericTrapezoid, Paraboloid
  - VecGeom shapes in Geant4 works in either batch or interactive mode
    - Supporting all features (multi-threading, parameterisations, scoring, etc..)
  - Ability to individually select shapes to replace at installation



(*) Full interactive Geant4 application with parallel geometries, parameterisations and scoring

- *Validation: Exact tracking response using hierarchical and realistic detector geometries (CMS, ALICE, LHCb) with ray-tracing (geantinos), compared to native Geant4*

VecGeom solids used through ROOT



WP3 milestones achieved

MILESTONE REPORT

RUNNING PROTOTYPE OF USOLIDS USING SIMD INSTRUCTIONS

MILESTONE: MS39

Document identifier:	AIDA2020-MS39
Due date of milestone:	End of Month 21 (January 2017)
Justification for delay	-
Report release date:	13/01/2017
Work package:	WP3: Advanced Software
Lead beneficiary:	CERN
Document status:	Draft

Abstract:

The USolids package [1] has been revised and extended to support vector signatures, providing the ability to perform queries on the geometrical primitives in parallel and allow for concurrent queries within the VecGeom package [2]. A running prototype of USolids within VecGeom is now available, implementing most of the shapes defining the standard set in the GDML schema [3]; SIMD (Single Instruction, Multiple Data) instructions are being used where possible, while the API has been extended to provide vector signatures. In this document, we briefly report on the design principles behind the prototype implementation.

MILESTONE REPORT

RUNNING PROTOTYPE FOR GEANT4 BASED SIMULATION TOOLKIT

MILESTONE: MS42

Document identifier:	AIDA2020-MS42
Due date of milestone:	End of Month 21 (January 2017)
Justification for delay	-
Report release date:	13/01/2017
Work package:	WP3: Advanced Software
Lead beneficiary:	CERN
Document status:	Draft

Abstract:

The USolids package [1] now available within VecGeom [2], provides an extended set of geometrical primitives with also vector signatures, which can be exercised through the Geant4 simulation toolkit [3]. In this document, we briefly report on the testing performed within Geant4 and the design behind the interface bridging Geant4 with the VecGeom package, such that original Geant4 primitives can be fully replaced in their functionalities with VecGom/USolids ones.

Next steps

- Progress in the implementation of the missing shapes from the standard set
  - Port of tessellated solid in VecGeom ongoing
  - Extruded solid, generic-polycone, etc…
  - Phase out old USolids module
- Progress in further extend testing coverage
  - On all shapes and their possible topologies
- Continue testing on realistic geometry setups
  - Robustness tests
  - Exact tracking comparison with original Geant4
  - Include new snapshots from LHC detectors
- Interface VecGeom navigation algorithms in Geant4 and ROOT

Summary

- VecGeom shapes ready to be used from either Geant4 and ROOT
- Production quality for a large set of primitives
- New solids implemented to complete the set existing in Geant4/ROOT/USolids
  - Still missing: half space (ROOT), full extruded solid (ROOT/Geant4), twisted shapes (Geant4)
- Re-designed interfaces for more flexibility, migration to VecCore vector library

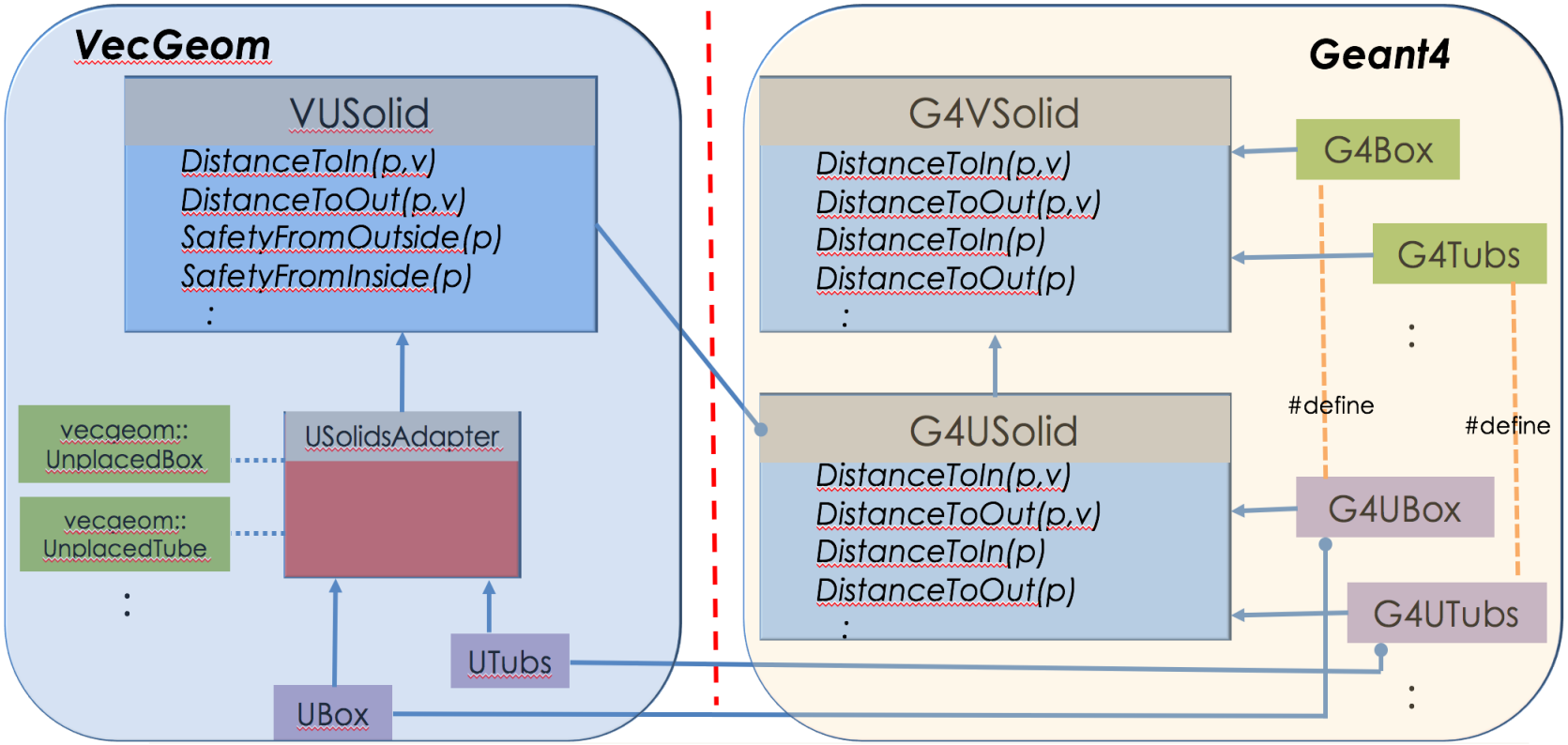
Contributors

- *CERN-EP/SFT + AIDA 2020: G.Amadio, J.Apostolakis, F.Carminati, G.Cosmo, A.Gheata, M.Gheata, W.Pokorski, E.Tcherniaev*
- *G.Lima (FNAL), R.Sehgal (BARC), S.Wenzel (CERN-ALICE)*

Backup

Using VecGeom solids in Geant4

- Wrappers in Geant4 (G4U*) making USolids look like native Geant4 solids
- Adapter in VecGeom wrapping either old USolids or new VecGeom solids behind the U* name.



Integration with ROOT