Detector Identifier and Geometry Management System in the JUNO Offline Software

Yuning Su[1], Chengxin Wu[1], Peidong Yu[2, 3], Ziyan Deng[2], Zhengyun You[1]
Jiaqi Lyu[1], Jiyang Zhao[1], Tao Lin[2], Yumei Zhang[1], Chenghao Tu[1]
suyn26@mail2.sysu.edu.cn

1. Sun Yat-sen University
2. IHEP
3. University of Chinese Academy of Sciences

24 May 2026
28th CHEP 2026, Bangkok

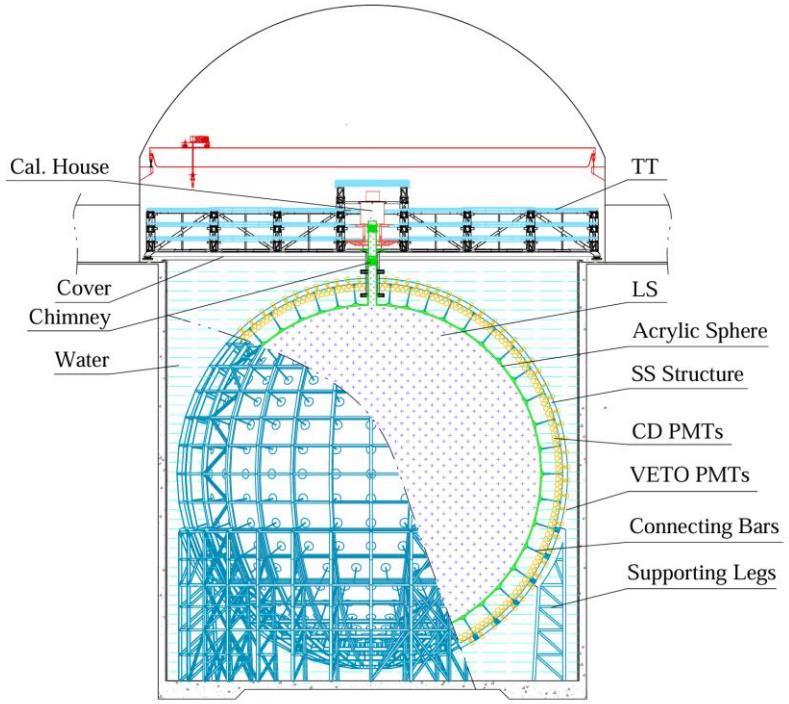
Outline

- **JUNO Experiment**
  - Introduction of JUNO Experiment
  - Status of JUNO Detector
  - Status of JUNO offline software
- **Identifier System in JUNO**
  - Introduction of Identifier
  - Introduction of Identifier Service
  - Workflow of Identifier Service
- **Geometry Management System in JUNO**
  - Introduction of Geometry Management
  - Design of Geometry Management System
- **Several applications in JUNO Offline**
  - Detector Simulation
  - Event display
  - Reconstruction and calibration
- **Summary**

Outline

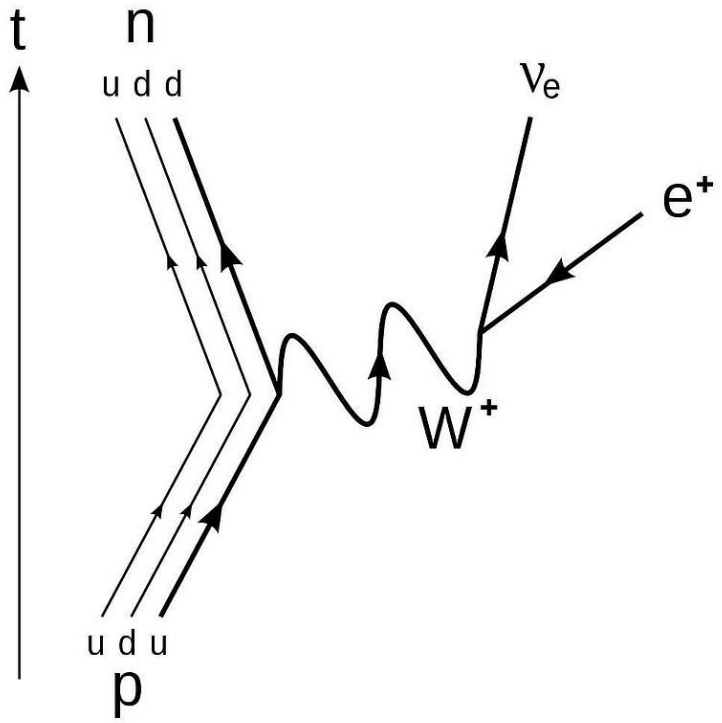
- **JUNO Experiment**
  - Introduction of JUNO Experiment
  - Status of JUNO Detector
  - Status of JUNO offline software
- **Identifier System in JUNO**
  - Introduction of Identifier
  - Introduction of Identifier Service
  - Workflow of Identifier Service
- **Geometry Management System in JUNO**
  - Introduction of Geometry Management
  - Design of Geometry Management System
- **Several applications in JUNO Offline**
  - Detector Simulation
  - Event display
  - Reconstruction and calibration
- **Summary**

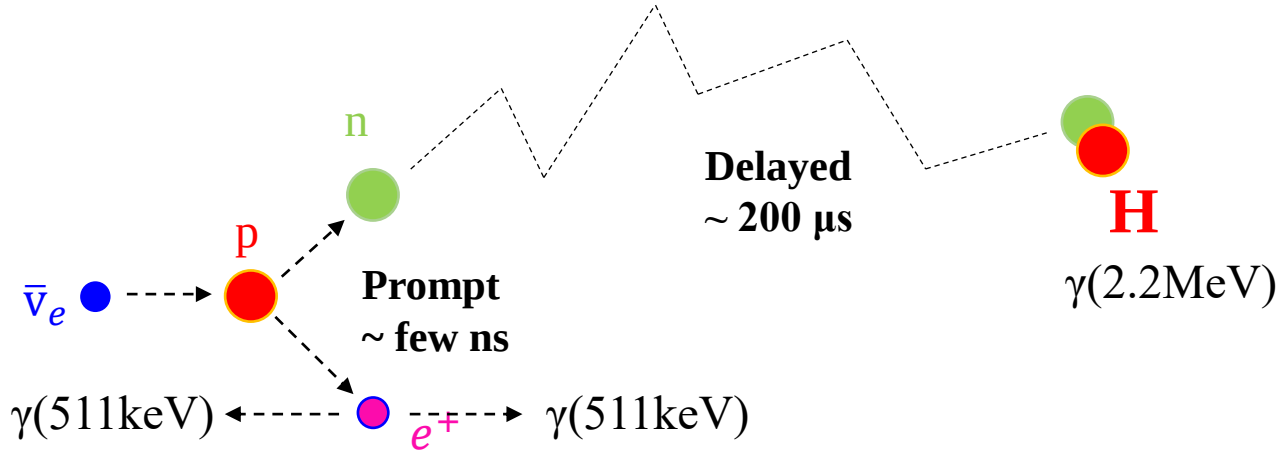
Introduction of JUNO Experiment



- **JUNO (Jiangmen Underground Neutrino Observatory)** is an international large-scale neutrino experiment located 700 meters underground in Kaiping, Guangdong Province, China.
- **Primary physics goals**:
  - Determine the neutrino mass ordering with a significance of 3~4 σ.
  - Measure neutrino oscillation parameters with unprecedented precision (up to 0.6%).
  - Search for physics beyond the Standard Model.

Introduction of JUNO Experiment

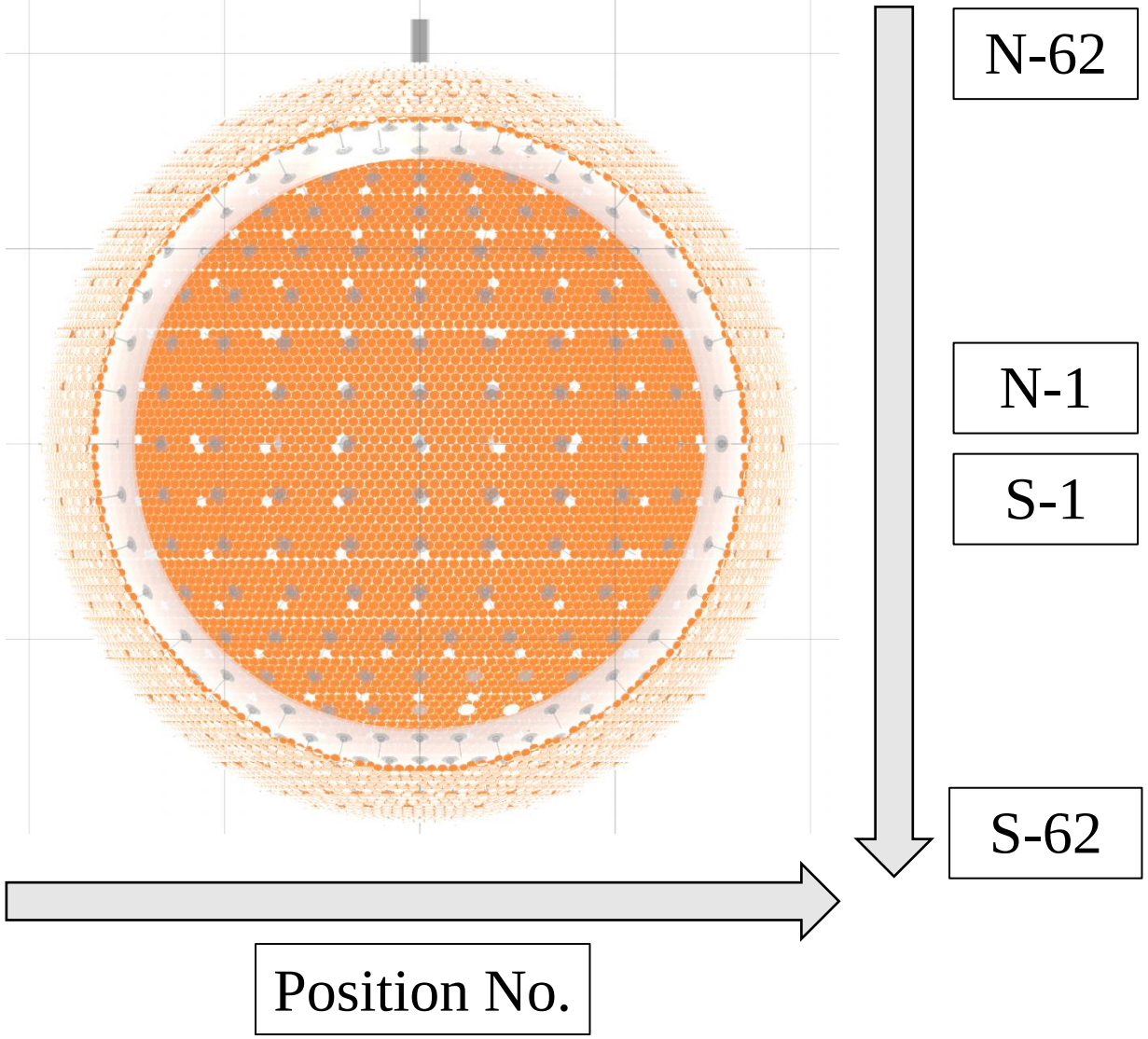




- JUNO detects neutrinos through the Inverse Beta Decay (IBD) process : $\bar{\nu}_e + p \rightarrow e^+ + n$
- Positron-electron annihilation → γ → **prompt signal (~ns)** ; neutron capture → γ → **delayed signal (~200μs)** ;
- Both signals are collected by **Photomultiplier Tubes (PMTs)** in the JUNO detector;

Status of JUNO Detector

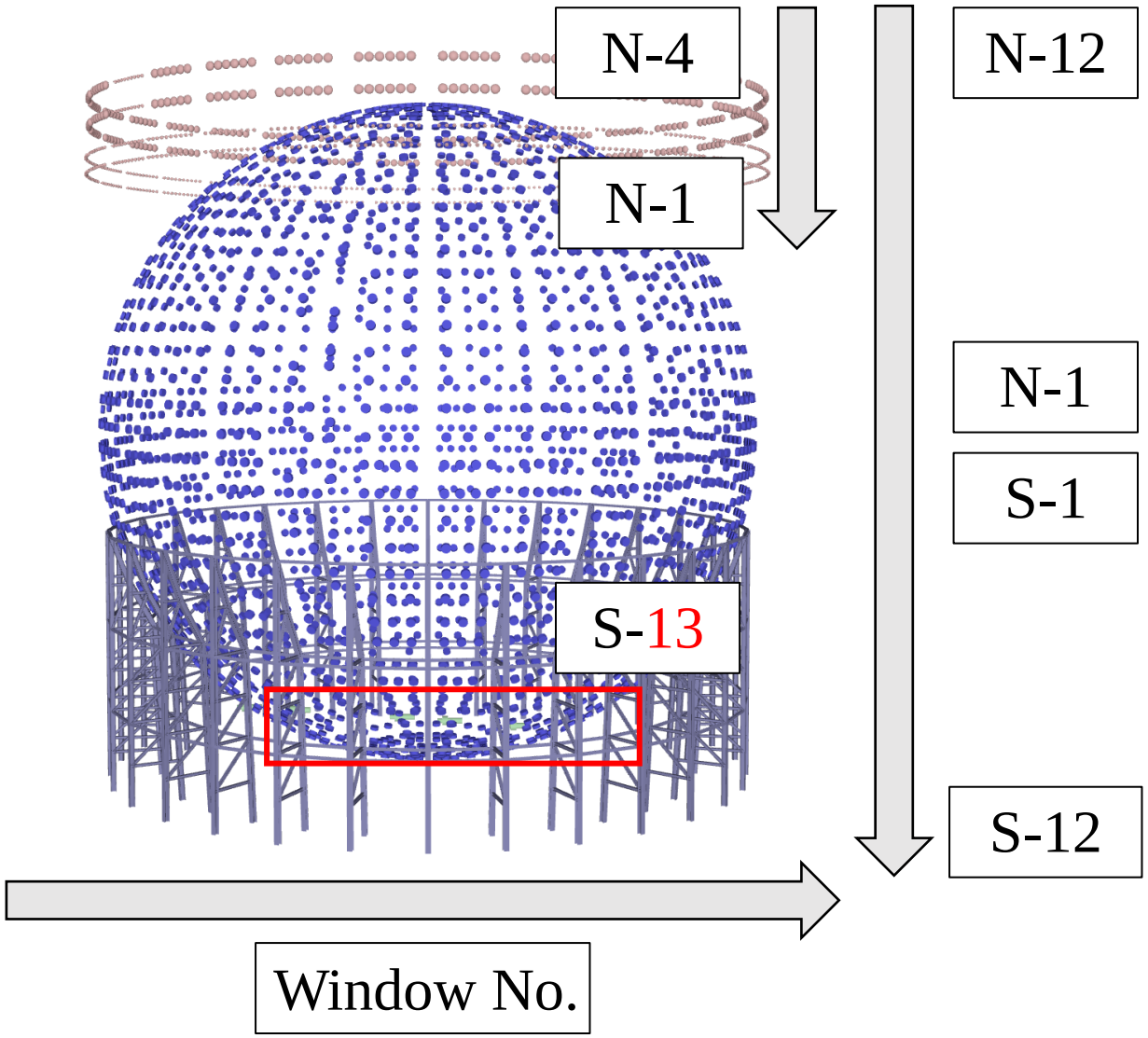
- The structure of Central Detector (CD).



- **Circle No.**
  - **124** in total: **62** in each hemisphere
- **Position No.**
  - Fewer PMTs near the poles (1~7)
  - More PMTs near the equator (1~195)
- **PMT type:**
  - NNVT/HighQENNVT (20 inch)
  - Hamamatsu (20 inch)
  - HZC (3 inch)

Status of JUNO Detector

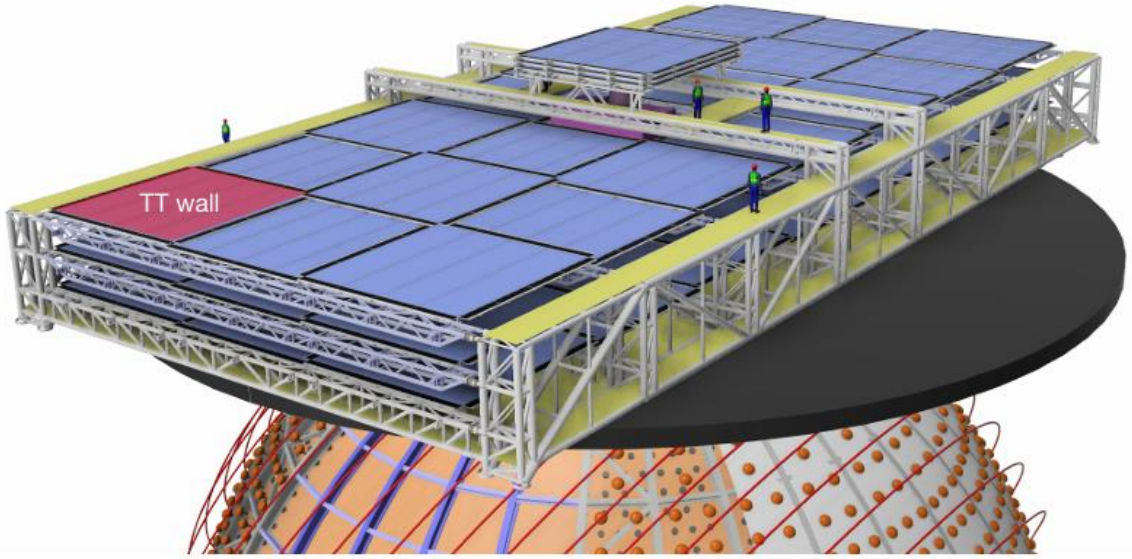
- The structure of Water Pool (WP).

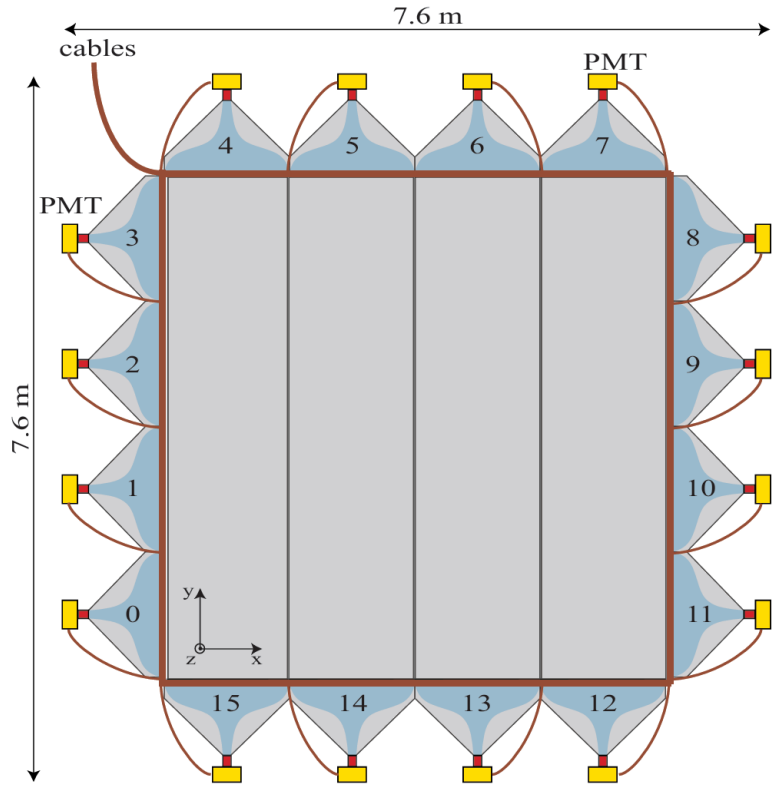


- **Layer No.**
  - **veto PMT: 24** in total, **12** in each hemisphere
  - **ATM PMT: 4** in total in the north hemisphere
  - **WAL PMT:** set as the **13th** layer
- **Window No.**
  - Fewer windows near the poles (1~10)
  - More windows near the equator (1~30)
- **Position No.**
  - **8** PMTs in each window (1~8)
- **PMT type:**
  - NNVT/HighQENNVT (20 inch)
  - Hamamatsu (20 inch)
  - 8-inch DYB (8 inch)

Status of JUNO Detector

- The structure of Top Tracker (TT).





layer → wall → plane → panel (module) → PMT 2

layer 3

wall 21(3×7) **62 in total**

plane 2

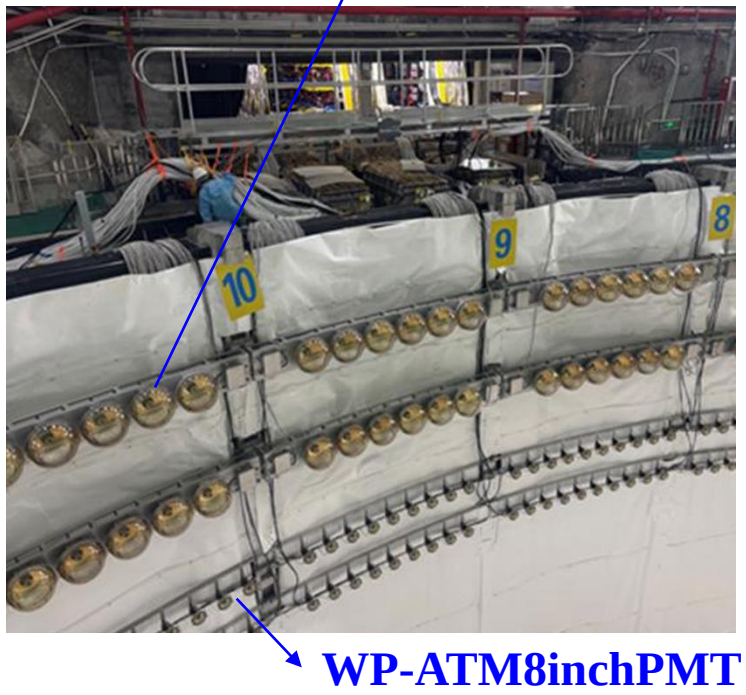
panel (module) 8 → channel 64 **31744 in total**

plastic scintillator strips

Status of JUNO Detector

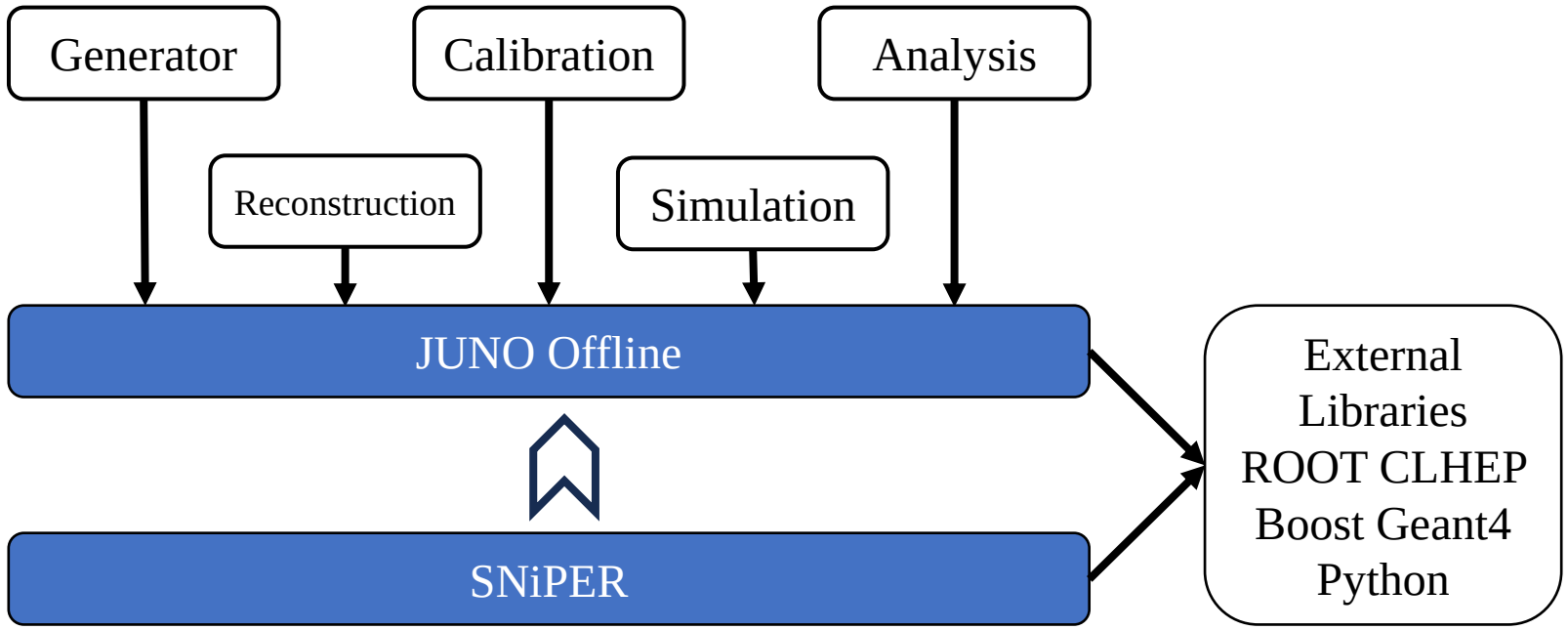
➢ Status of PMTs in JUNO :

- **17596** CD-LPMT (20-inch), **25600** CD-SPMT (3-inch), **2399** WP-VPMT (20-inch), **5** WP-WALPMT (20-inch) , **348** WP-ATM-LPMT (20-inch) and **600** WP-ATM-MPMT(8-inch) **are installed**.



Status of JUNO Offline Software

- Established in 2012, the JUNO offline software (JUNOSW) has been developed to fulfill various demands of JUNO :
  - **Core Software:** Framework, Event Data Model, Geometry Management, Database System, Analysis Framework
  - **Applications:** simulation, calibration, reconstruction and analysis
  - **Modern management technologies:** C++17, python3, Cmake, Gitlab



Outline

- **JUNO Experiment**
  - Introduction of JUNO Experiment
  - Status of JUNO Detector
  - Status of JUNO offline software
- **Identifier System in JUNO**
  - Introduction of Identifier
  - Introduction of Identifier Service
  - Workflow of Identifier Service
- **Geometry Management System in JUNO**
  - Introduction of Geometry Management
  - Design of Geometry Management System
- **Several applications in JUNO Offline**
  - Detector Simulation
  - Event display
  - Reconstruction and calibration
- **Summary**

Introduction of Identifier

- Identifier (**32-bit unsigned integer** 0x00000000) can :
  - provides a unique identification number for every detector unit with a readout and designed for different sub-detectors (**CD, WP and TT**)
  - builds correlations between data and detector units
  - serve as a unified framework for global access

- **The Identifier** has been developed as a well-structured ID segment **since early JUNOSW version**, and **won't change anymore** :
  - It is constructed relying on the hierarchical position and type information of PMTs.
  - The **number** of PMTs and Identifiers **are fixed**, based on the csv data files saved in JUNO database.
  - Users **should not** generate Identifiers by themselves, use Identifier service to access Identifier instead.
  - The Identifiers of uninstalled PMTs can be retrieved in Identifier Service as well.

Introduction of Identifier (Exp: CD)

Sub-Detector: 1

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Sub-Detector (0~15)				L/S	N/S	Circle No. (6bits 0~63)						Position No. (0~255)			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Position No. (0~255)				U/D	PMT Type										

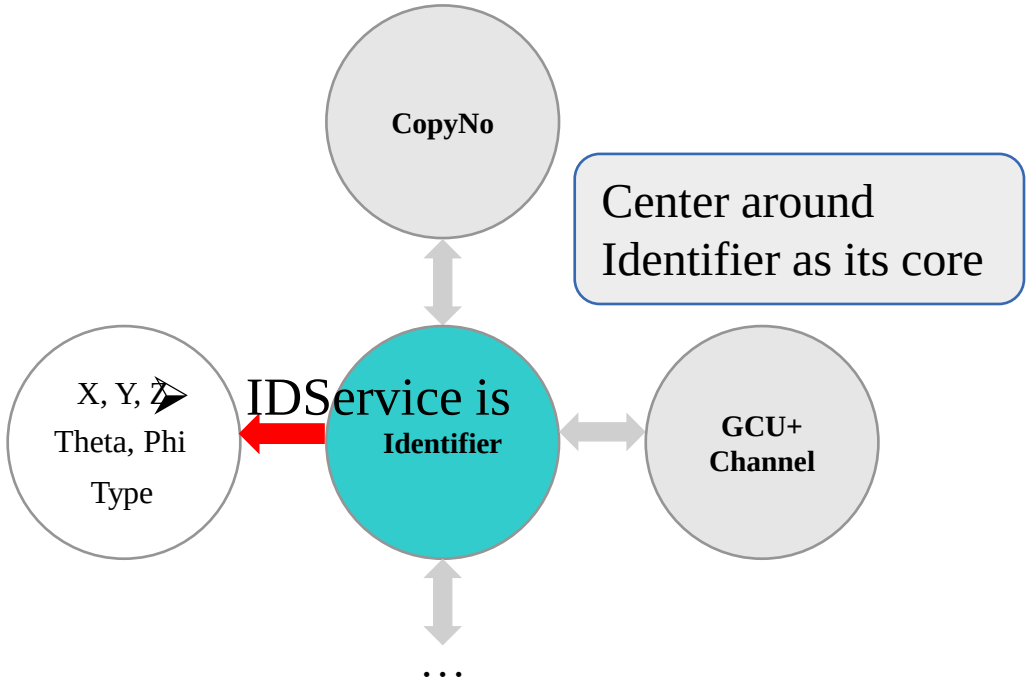
- L/S: Large (0) / Small (1) PMT
- N/S: North (0) / South (1) in CD
- Circle No.: 0~63
- Position No.: 0~255
- U/D: Up (0) / Down (1), **small PMT only**, relative to its reference large PMT
- PMT Type: Hamamatsu (1), NVVT_1 (2), NVVT_2 (3), Small (4)

Example of a large PMT in CD: 1-L-N-07-152-0-1

a small PMT in CD: 1-S-N-07-152-**1**-4

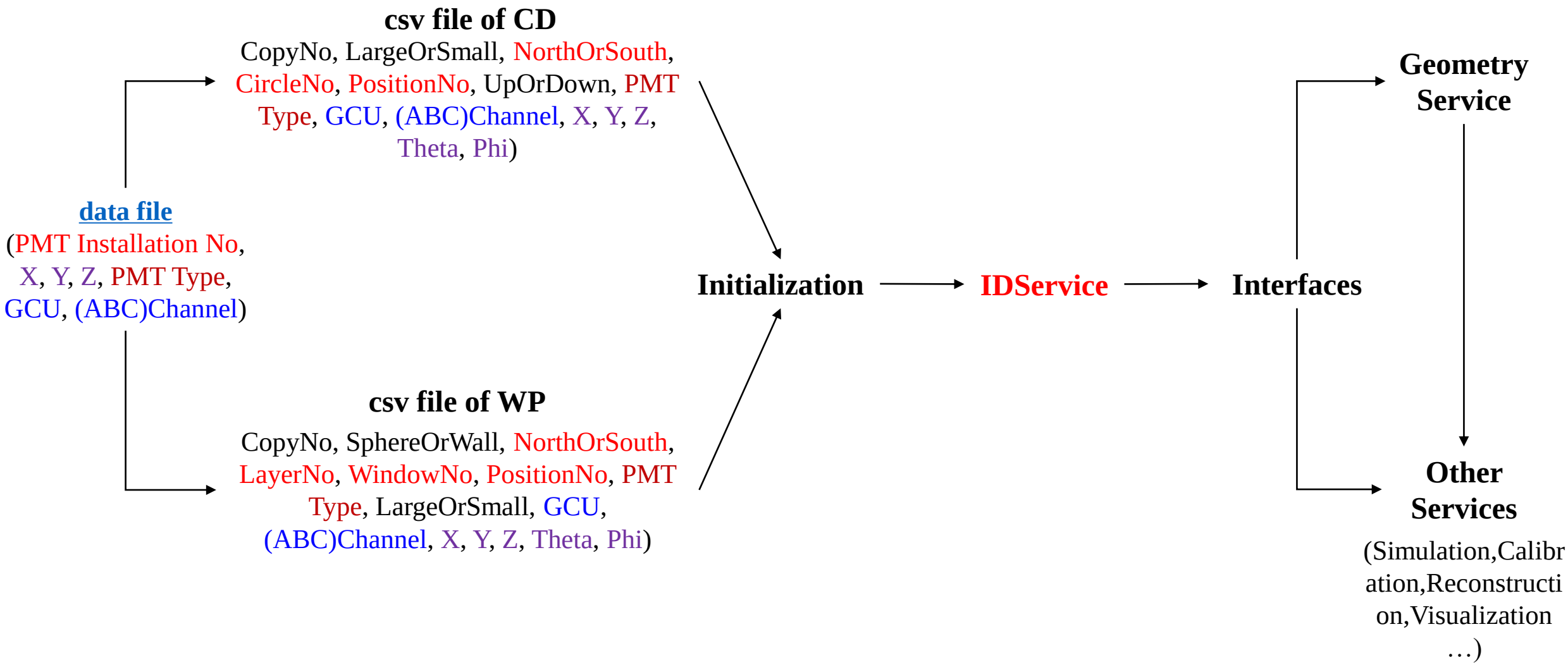
Introduction of Identifier Service

- Various applications (offline, DAQ, Electronic box, Install-ID, etc.) have their own numbering rules to distinguish each PMT.
- ID Mapping Service (**IDService**) provides correlations between different ID systems and between ID systems and geometric information.



- **IDService is initialized :**
  - To Generate **Identifier** for each detector unit (PMT)
  - To construct the **bi-directional** correlation mapping between Identifier and different **ID systems** :
    CopyNo ↔ Identifier (complete)
    GCU + Channel ↔ Identifier (complete for installed PMTs)
  - To construct the **single-directional** correlation mapping between Identifier and **geometric information**:
    Identifier → x, y, z … (complete)
  - Outside the SNiPER framework and support multiple methods to access

Workflow of Identifier Service



Outline

- **JUNO Experiment**
  - Introduction of JUNO Experiment
  - Status of JUNO Detector
  - Status of JUNO offline software
- **Identifier System in JUNO**
  - Introduction of Identifier
  - Introduction of Identifier Service
  - Workflow of Identifier Service
- **Geometry Management System in JUNO**
  - Introduction of Geometry Management
  - Design of Geometry Management System
- **Several applications in JUNO Offline**
  - Detector Simulation
  - Event display
  - Reconstruction and calibration
- **Summary**

Introduction of Geometry Management

- The **geometry management system** is essential for :
  - providing complete spatial parameters of the detector accurately.
  - supporting conversions between different geometry formats.
  - defining the physical dimensions and spatial layout.

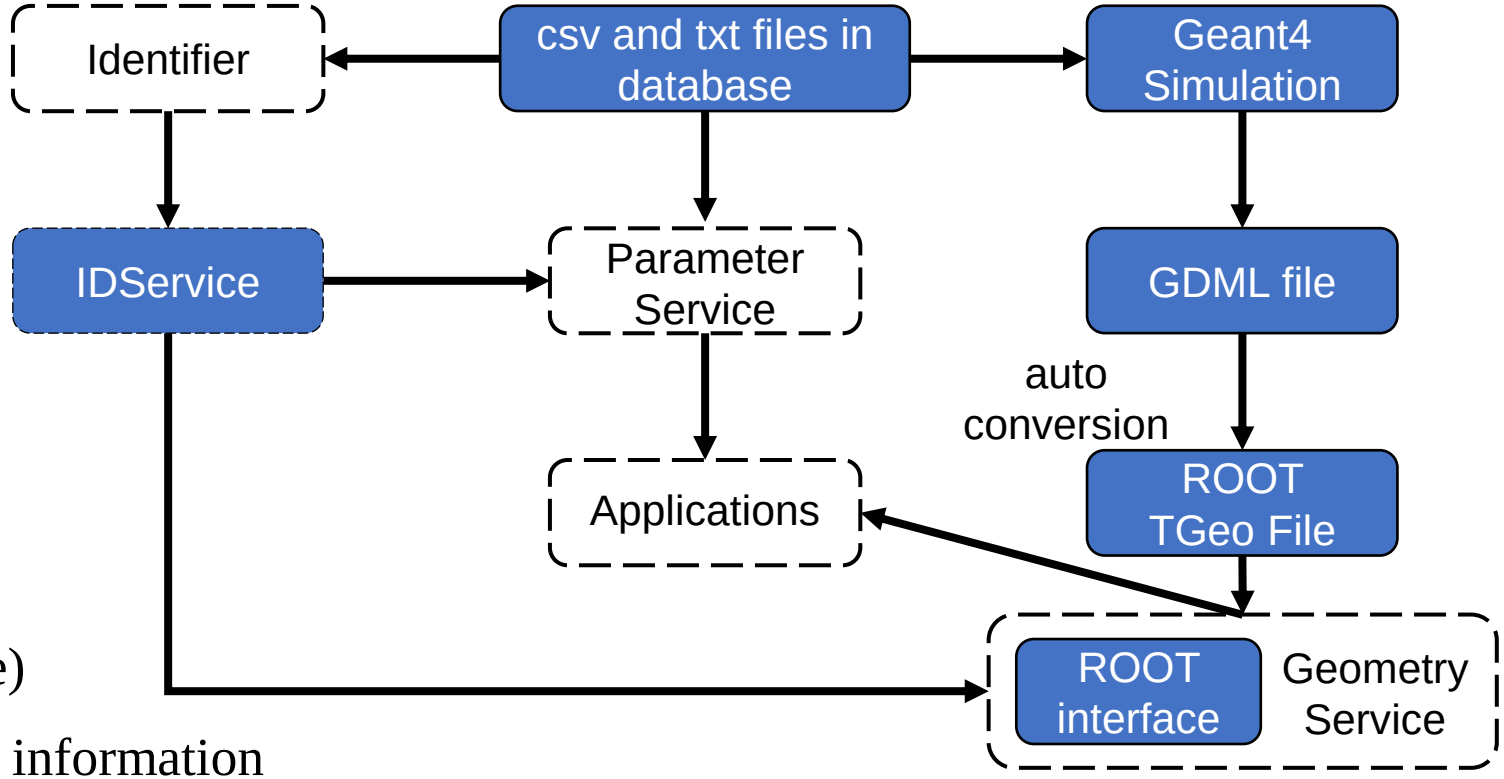
- It is based on a **detector description system**, which acts as the central source of detector information.

- The detector description system is implemented by multiple frameworks: **Geant4**, **GDML (XML based)**, **ROOT**

Design of Geometry Management System

- Geometry management of JUNO is designed as a **dual-geometry system**: consists of parameter service and geometry service system.

- Simple geometric structure (parameter service)
  - **Quickly** access key detector geometry information
  - Aimed at **reducing** memory consumption



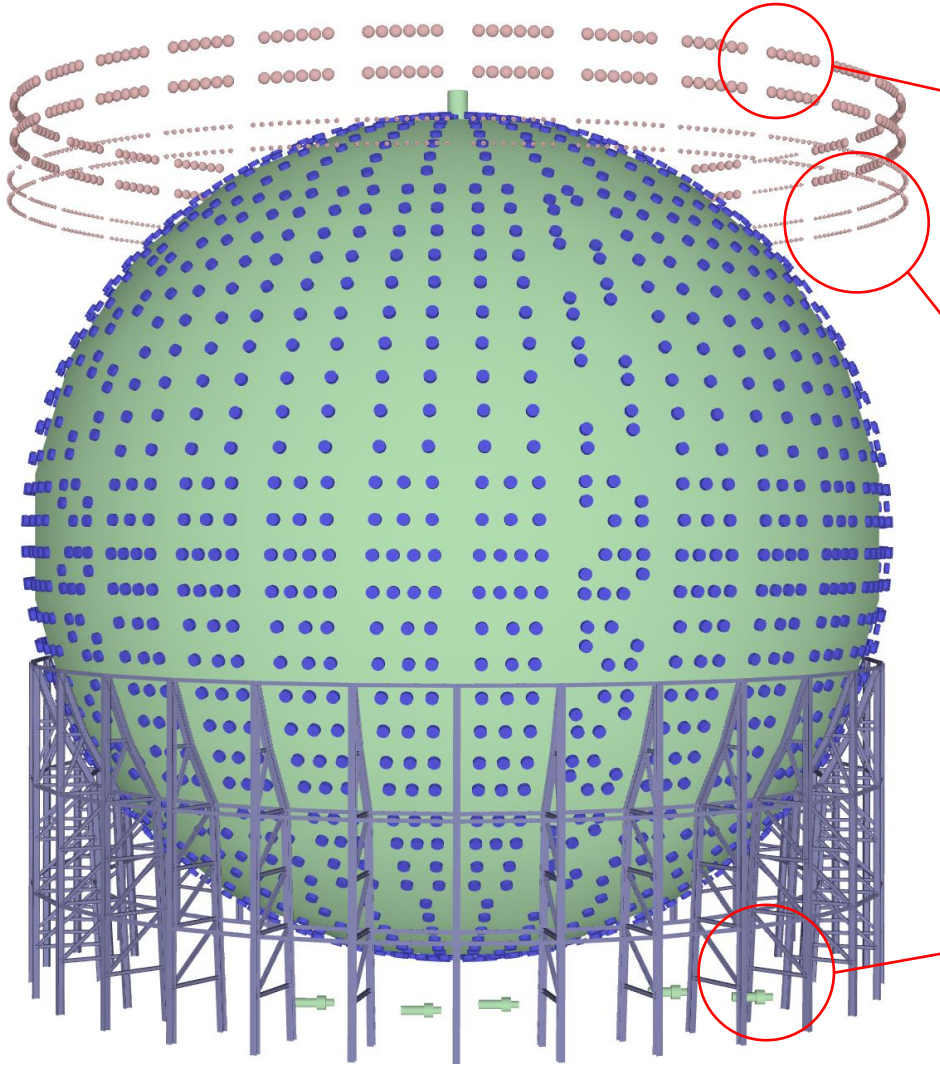
- Full geometric structure (geometry service)
  - Provide access to all **detailed** detector information
  - Offer a comprehensive description of the detectors
  - Initialized with GDML from GDML file exported in simulation

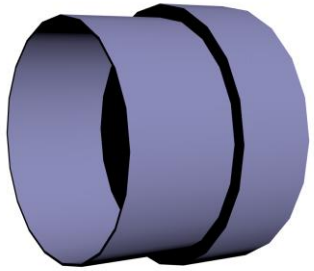
Outline

- **JUNO Experiment**
  - Introduction of JUNO Experiment
  - Status of JUNO Detector
  - Status of JUNO offline software
- **Identifier System in JUNO**
  - Introduction of Identifier
  - Introduction of Identifier Service
  - Workflow of Identifier Service
- **Geometry Management System in JUNO**
  - Introduction of Geometry Management
  - Design of Geometry Management System
- **Several applications in JUNO Offline**
  - Detector Simulation
  - Event display
  - Reconstruction and calibration
- **Summary**

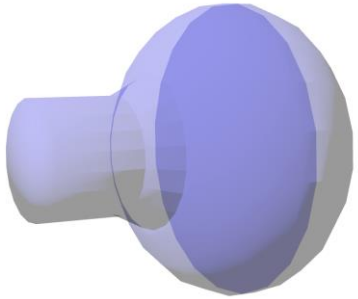
Detector Simulation

➢ **Detector Simulation :**

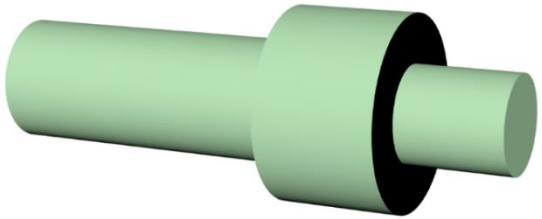




ATM-20inch PMT



ATM-8inch PMT



WP WAL PMT

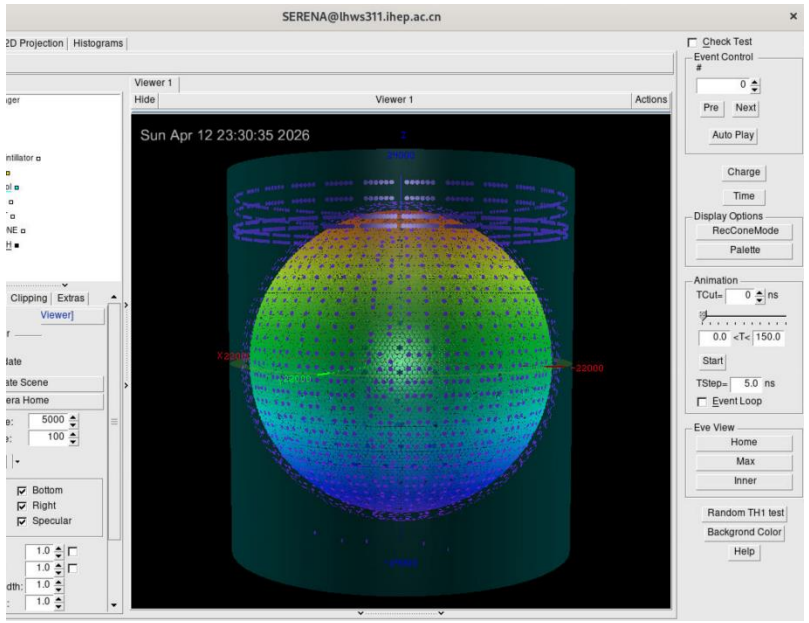
➢ The Identifier and geometry management system could help :

- load PMT parameters via SNiPER
- read PMT geometric and type information
- build position and rotation transformations for Geant4 construction
- control PMT installation status

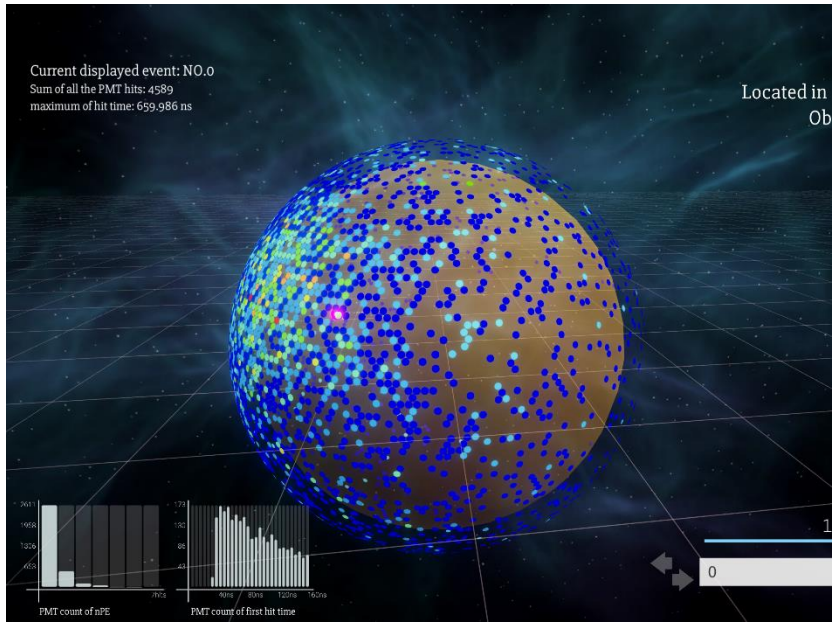
Event Display

➢ **Event Display :**

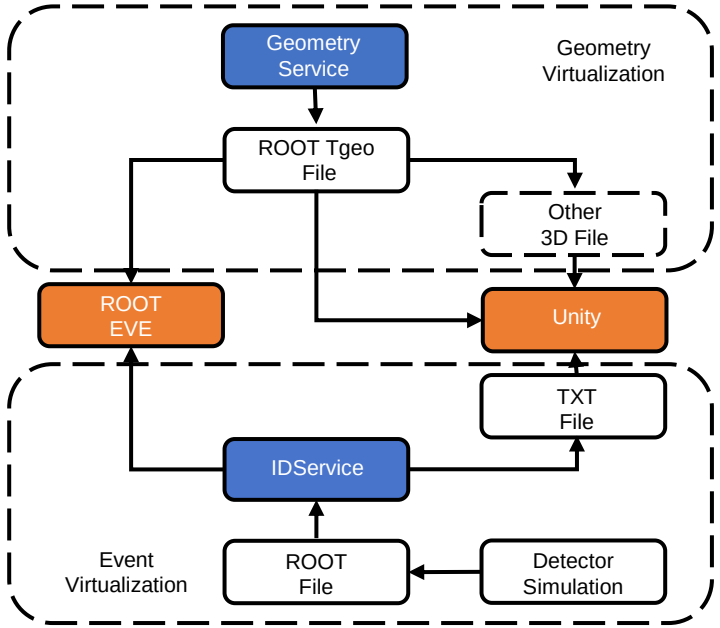
- It provides : **efficient visualization of detector geometry**, **tracks and hit patterns**, facilitating data monitoring, algorithm validation, and the interpretation of complex physics processes.
- the mainstream visualization technologies for HEP experiments include **ROOT-based** visualization and **Unity-based** visualization.



ROOT EVE



Unity

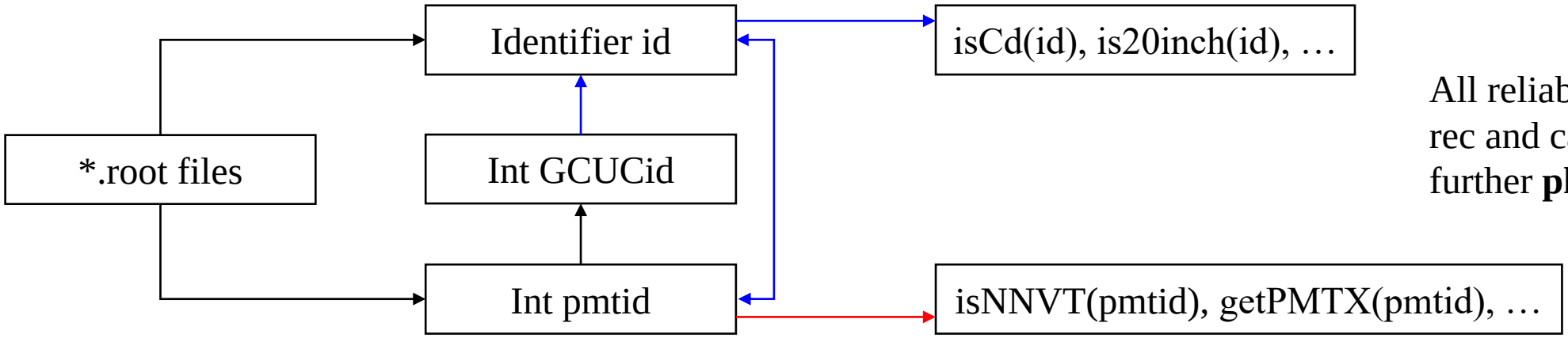


Workflow of ROOT and Unity-based Visualization

Reconstruction and calibration

➢ **Reconstruction and Calibration :**

- Implements the **IDService** to :
  - convert Identifier of PMTs into integer-type copy number for **array indexing, classification** or **validation**.
  - convert pmtid / GCUC into Identifier for **filtering** or any other purposes.
- Implements the **Geometry Service** to :
  - access numerous functional methods, including distinguishing the type of PMTs, counting total number, getting PMT position, etc.



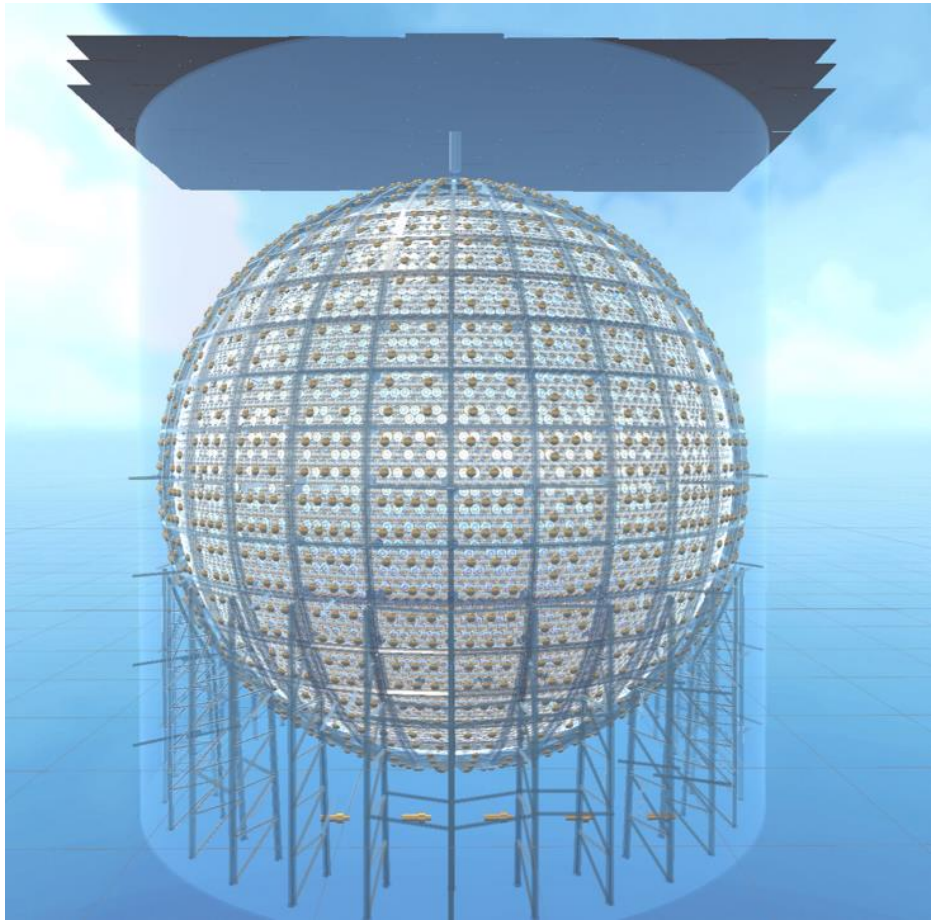
All reliable outcomes from rec and calib will be used in further **physics analysis!**

Outline

- **JUNO Experiment**
  - Introduction of JUNO Experiment
  - Status of JUNO Detector
  - Status of JUNO offline software
- **Identifier System in JUNO**
  - Introduction of Identifier
  - Introduction of Identifier Service
  - Workflow of Identifier Service
- **Geometry Management System in JUNO**
  - Introduction of Geometry Management
  - Design of Geometry Management System
- **Several applications in JUNO Offline**
  - Detector Simulation
  - Event display
  - Reconstruction and calibration
- **Summary**

Summary

- **JUNO** is an international large-scale neutrino experiment featuring several core physics targets.
- The main detector system comprises the CD and WP, equipped with **43196** CD-PMTs and **3352** WP-PMTs in total.
- Identifier serves as a **unique ID** system and initialized by PMT information.
- Identifier Service provides **mapping correlations** for both ID systems and geometric properties of PMT.
- The geometry management system is a **dual-geometry** system with two services for different purposes.
- The detector Identifier and geometry management system have been successfully implemented in **JUNO offline software** including **detector simulation**, **visualization**, **calibration** and **reconstruction**, given great support to **physics analysis**.



Thanks for your listening!

Backup

Design of Identifier (WP)

☐ **Sub-Detector: 2**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Sub-Detector (0~15)				N/S	Layer No. (0~15)				Window No. (0~31)					Position No.	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
(0~15)		PMT Type (0~3)													

☐ WP-Veto PMT :
- WP-Sphere-**South/North**-(1~12)-(1~30)-(1~8)-Type

☐ WP-VAN PMT :
- WP-Sphere-**South**-**13**-1-(1~5)-Type

☐ WP-ATM PMT **:**
- WP-**Wall**-**North**-(1~4)-(1~30)-(1-6)-Type

☐ WP-ATM PMT (Wall **348** LPMTs; **600** SPMTs)
- maintain compatibility with the previous design.

31	30	29	28
S/W		1	0

☐ **Sub-Detector: 2**
☐ **S/W: Sphere(0)** or **Wall(1)** (is VAN PMT)
☐ **Avaliable**

11
L/S

☐ L/S: Large(0) /Small(1) (20inch or 8inch) different from CD-SPMT(3inch)

- For the PMTs on the **CD sphere**, they are assumed to be **Large** PMTs by default.

Design of Identifier (TT)

☐ Sub-Detector: 3

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Sub-Detector (0~15)				Wall No.											

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
						Channel No.									

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Sub-Detector (0~15)				Layer No.											

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Column No.			Row No.			PMT No.				Strip No.					

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Sub-Detector (0~15)				Wall No.											

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
						PMT No.				Strip No.					

Usage for IDService

- Inside JUNO offline with C++ (exp: IDService_Test) :
  - `#include "Identifier/IDService.h" // add head file`
  - `IDService* idServ = IDService::getIdServ(); // create a object`
  - `idServ -> init();  // initialize it`
  - `idServ -> copyNo2Id(CopyNo); // from CopyNo to Identifier`
  - ……

- Outside JUNO offline with C++ (exp: testIDS.C, more details: DocDB-11158) :
  - `#include “Identifier/IDService.h” // add head file`
  - `Int_t status = gSystem -> Load("libIdentifier.so"); // load the library`
  - `IDService* idServ = IDService::getIdServ(); // create a object`
  - `idServ -> init();  // initialize it`
  - ……

Usage for IDService

- Inside JUNO offline with python (more detail: binding and share) :
  - `Import Sniper // import Sniper`
  - `Sniper.loadDll("libIdentifier.so") // load the library`
  - `s = libIdentifier.GCUC2Id(GCU, channel) // from GCUC to Identifier`
  - `id = int(s,16) // transformation`

- Outside JUNO offline with python (exp: Get_PMT_ID.py) :
  - `Import ctypes // import ctypes`
  - `Import ROOT // import ROOT`
  - `ROOT.gSystem.Load("libIdentifier,so") // load the library`
  - `ROOT.gInterpreter.Declare(‘”#include‟Identifier/IDService.h‟”’) // add the head file`
  - `idServ = ROOT.IDService.getIdServ() // create a object`
  - `idServ.init() // initialize it`
  - `identifier = id_serv.copyNo2Id(CopyNo) // from CopyNo to Identifier`

Usage for PMT Parameter Service

- Simple usage (exp: Geo_test.C) :
  - `#include “Geometry/PMTParamSvc.h” // add head file`
  - `PMTParamSvc* pmtsvc = new PMTParamSvc("PMTParamSvc"); // create a object`

- Sniper-Based usage :
  - `#include “Geometry/IPMTParamSvc.h” // add head file`
  - `SniperPtr<IPMTParamSvc> pmtsvc(getParent(), "PMTParamSvc"); // create a object`
  - `m_pmtParamSvc = pmtsvc.data(); // obtain data`