# The History of the Accept and Rise of Geant4

ISGC 2023, Taipei

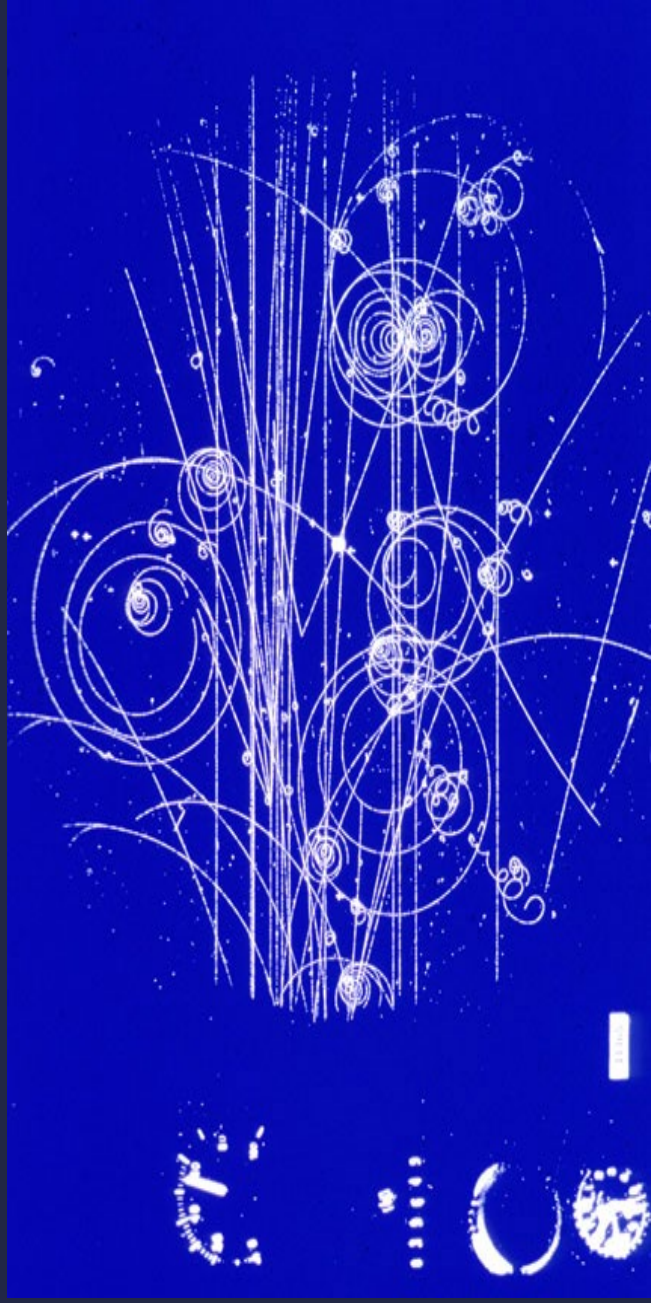
Takashi Sasaki/KEK

# Happy retirement, Simon!

# Congratulations on the 20th anniversary of ISGC!

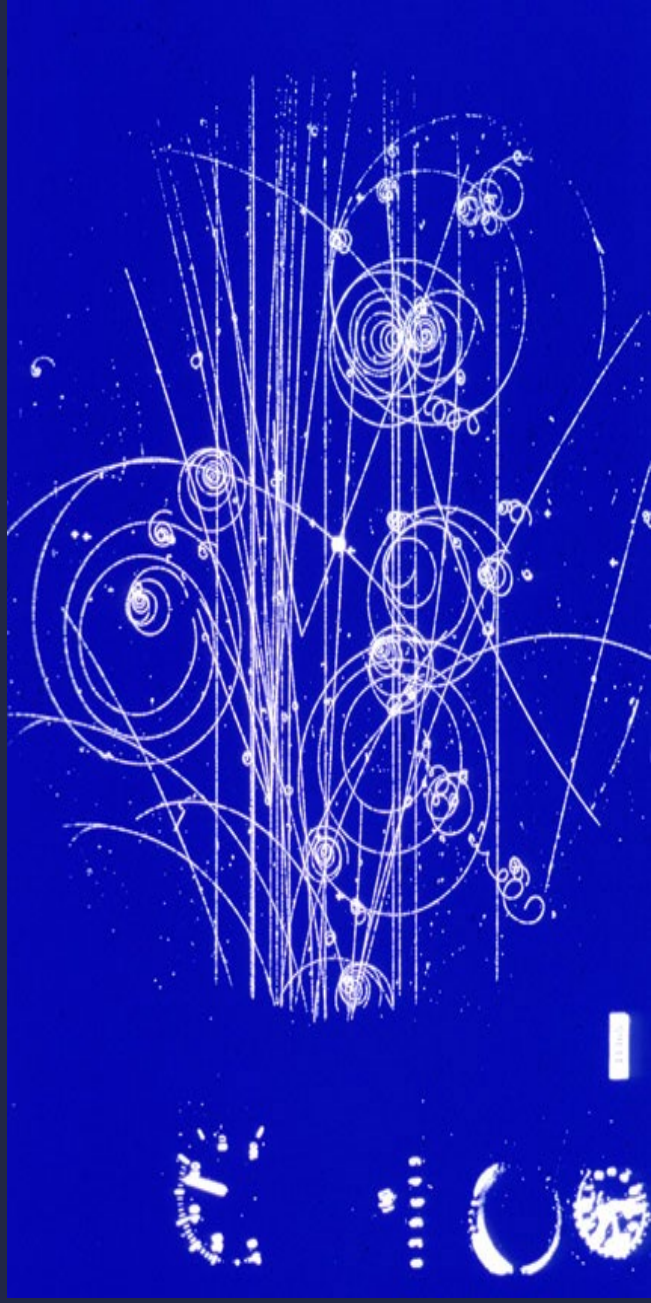
# Disclaimer

- I'm talking about my personal experience in Geant4
  - Something behind the official documentation
  - This is not an official presentation of the Geant4 collaboration
- Some parts of this presentation are courtesy of other Geant4 developers and users



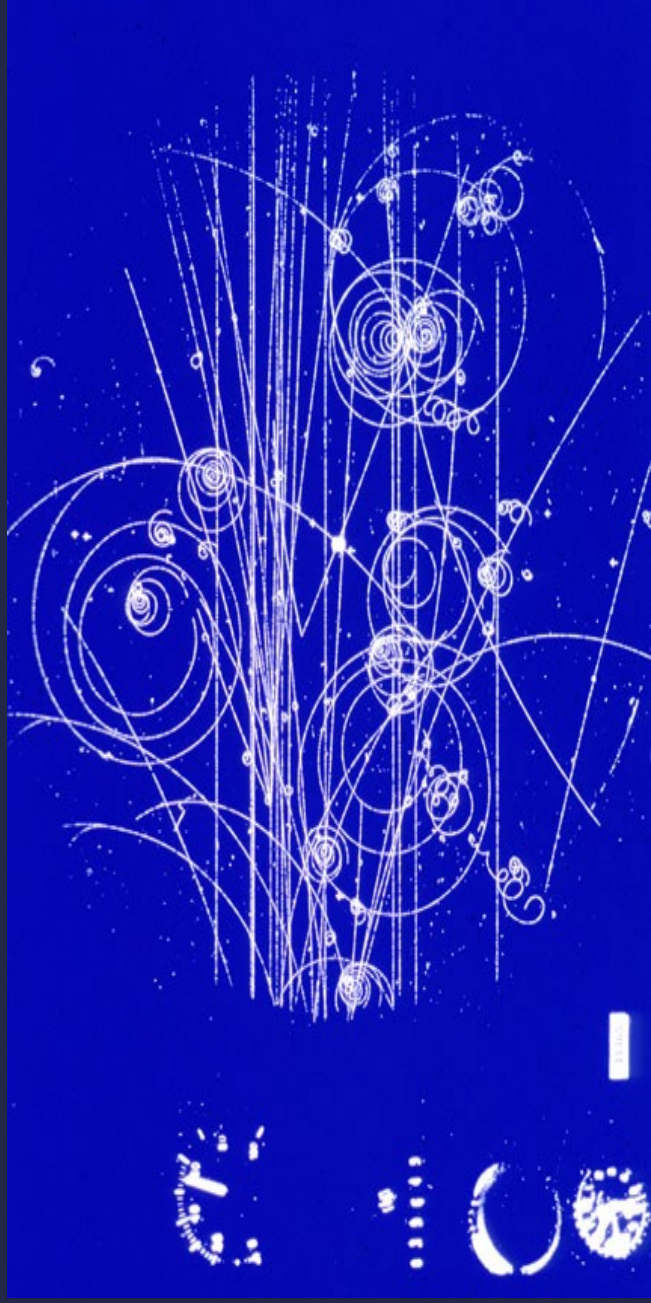
# What is Geant4?

- Started the development in 1994 based on earlier Japanese efforts
- Deployed Object-Oriented technologies fully, OOA/OOD/OOP
- A software toolkit to simulate interactions between particles and matter
  - Radiation transport simulation software
  - https://www.geant4.org/
- The biggest resource eater on the GRID sites
  - Half of the computing resources in WLCG are consumed by the defector simulation based on Geant4
  - All ongoing HEP experiments use Geant4 for their detector simulation currently

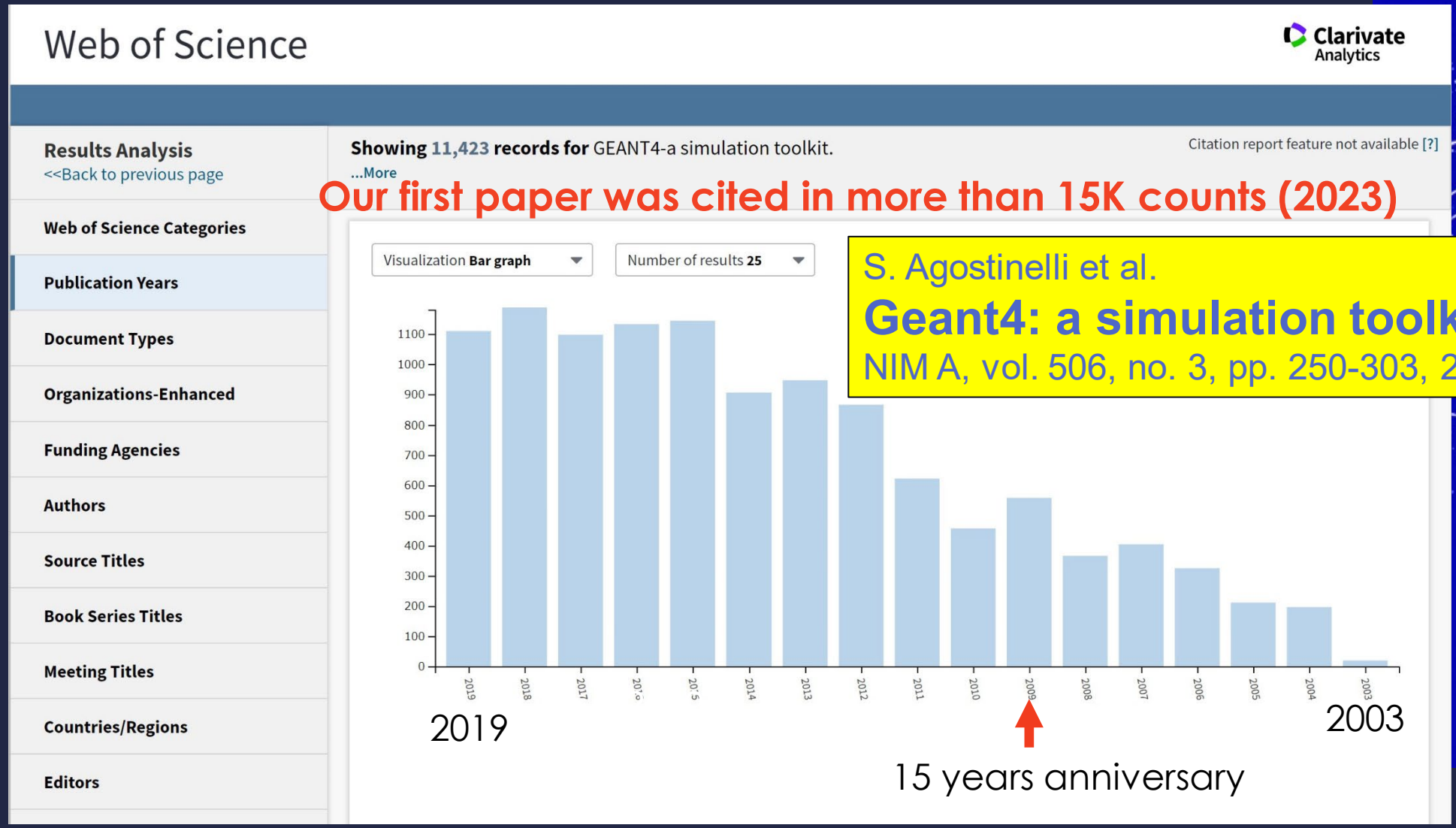


# Geant4 License

- Geant4 is an open-source software
- Rather relaxed license
  - Allows commercial use
  - Essence from the BSD license
- https://geant4.org/download/license.html
  - Lots of individual Japanese names on the page
  - Why?
    - You will learn the reason soon



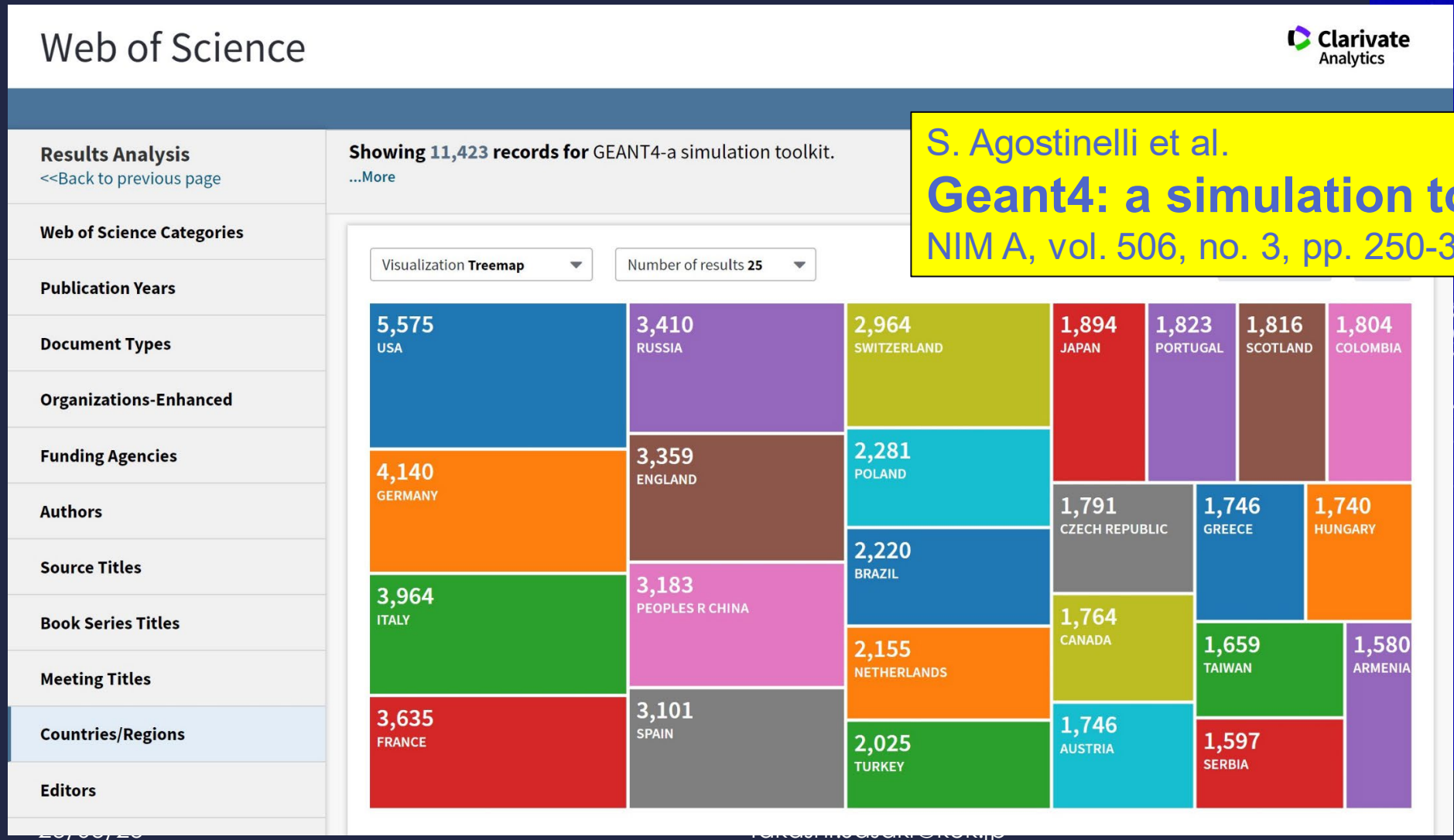
# The most popular radiation simulator




S. Agostinelli et al.
**Geant4: a simulation toolkit**
NIM A, vol. 506, no. 3, pp. 250-303, 2003

2003

# Worldwide recognition

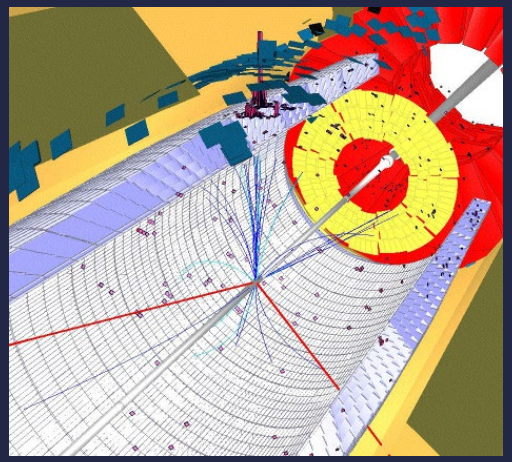


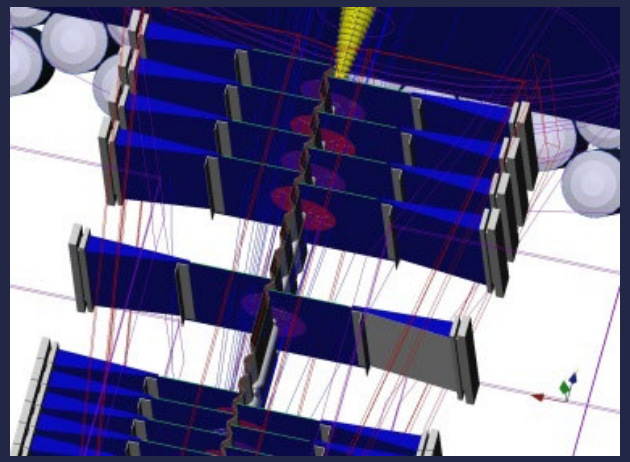

S. Agostinelli et al.

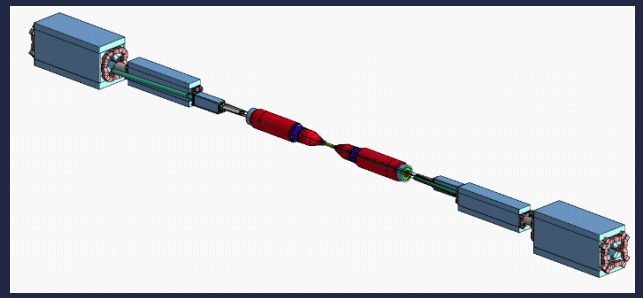
**Geant4: a simulation toolkit**

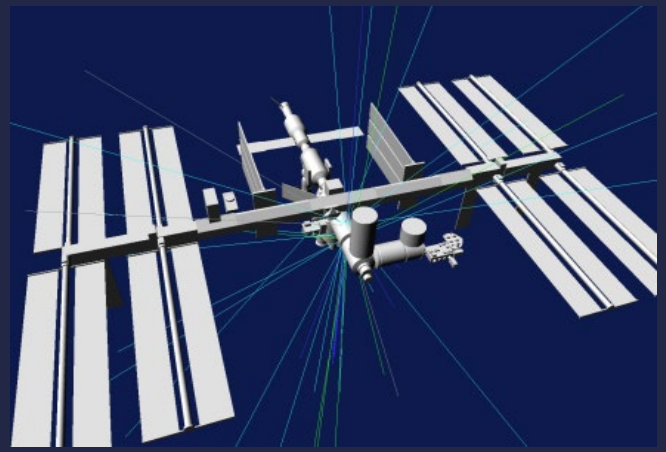
NIM A, vol. 506, no. 3, pp. 250-303, 2003

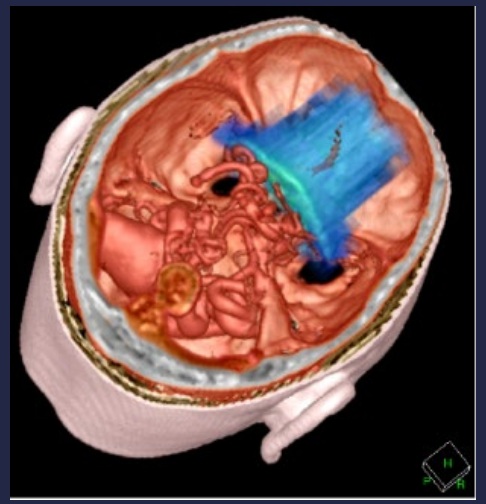
# Multi-discipline



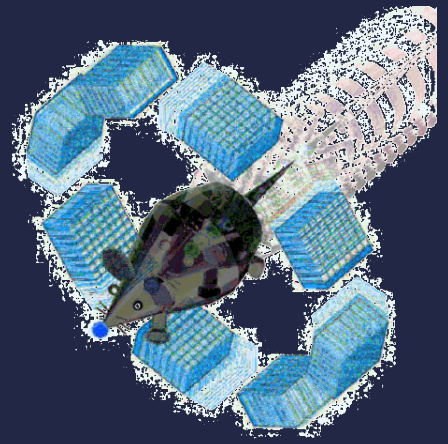


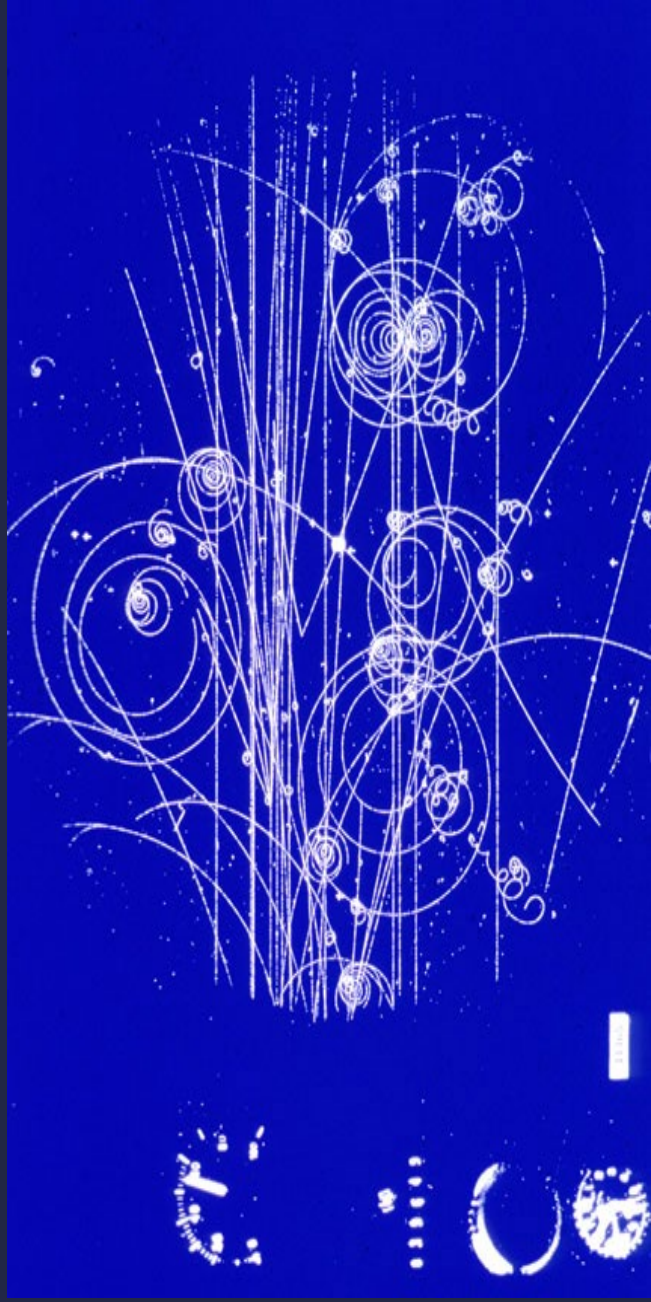




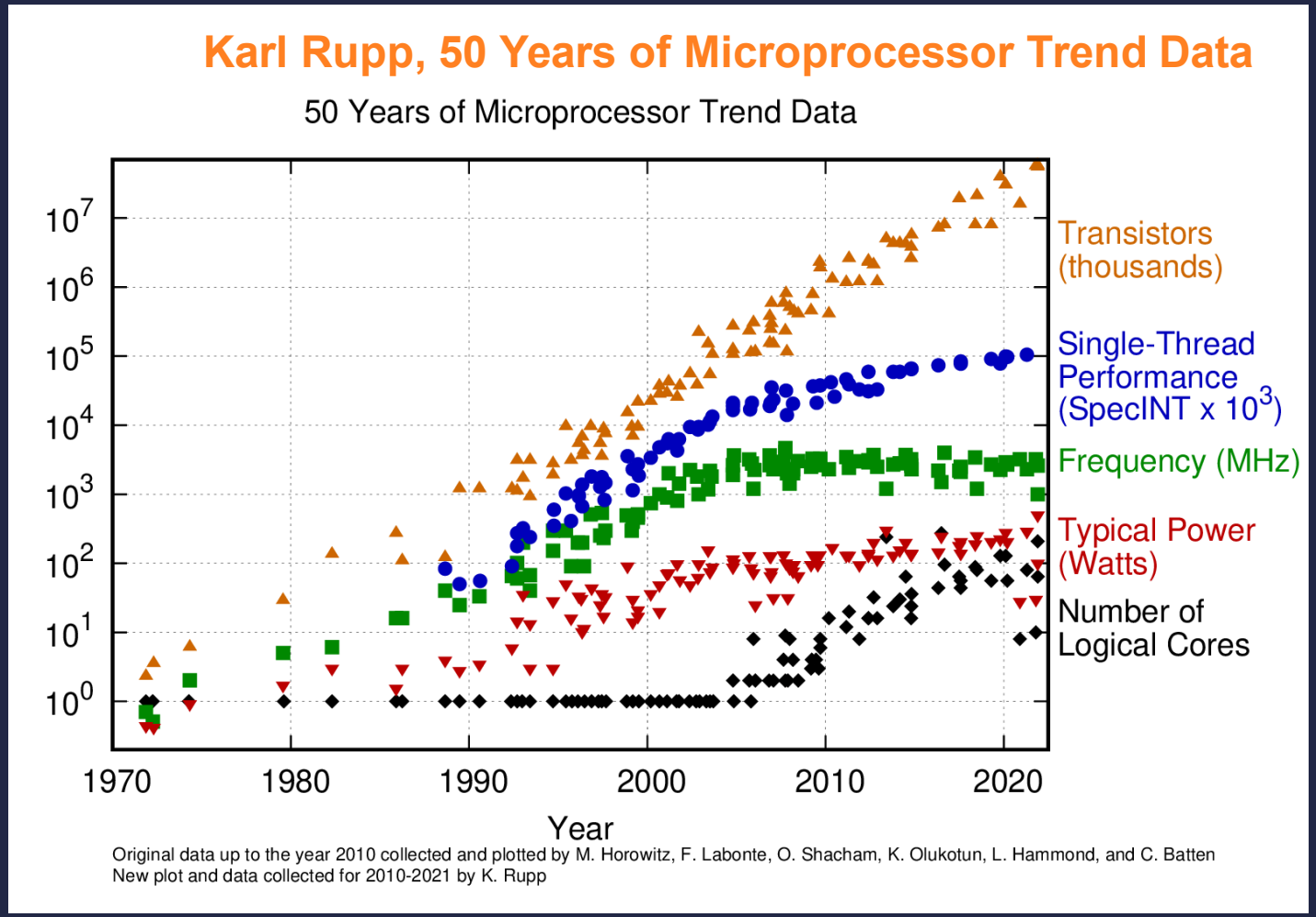


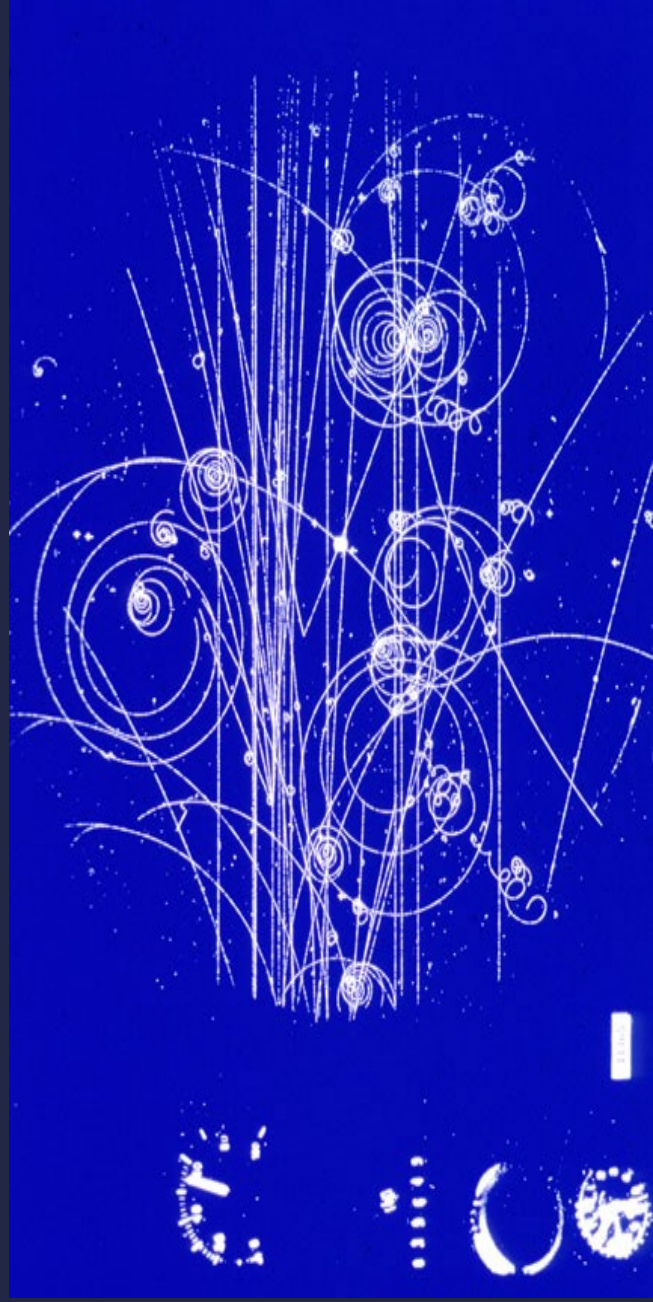
Courtesy T. Ersmark, KTH Stockholm





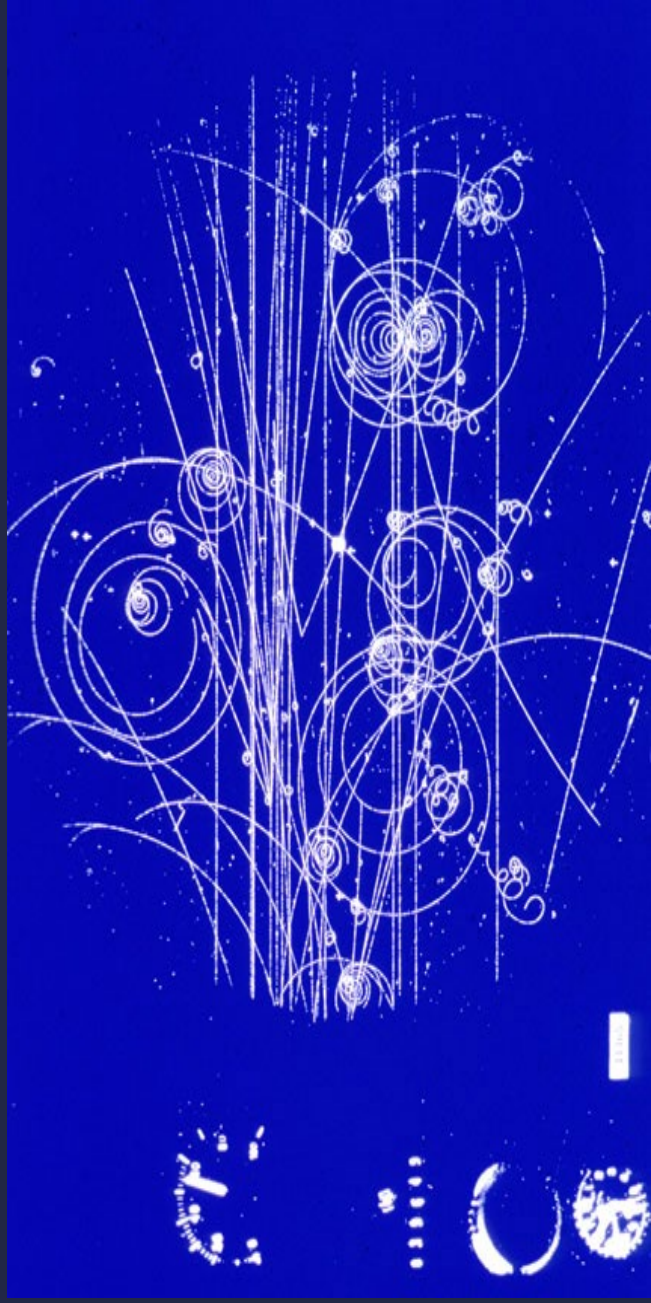
# Evolution continues following microprocessor trend



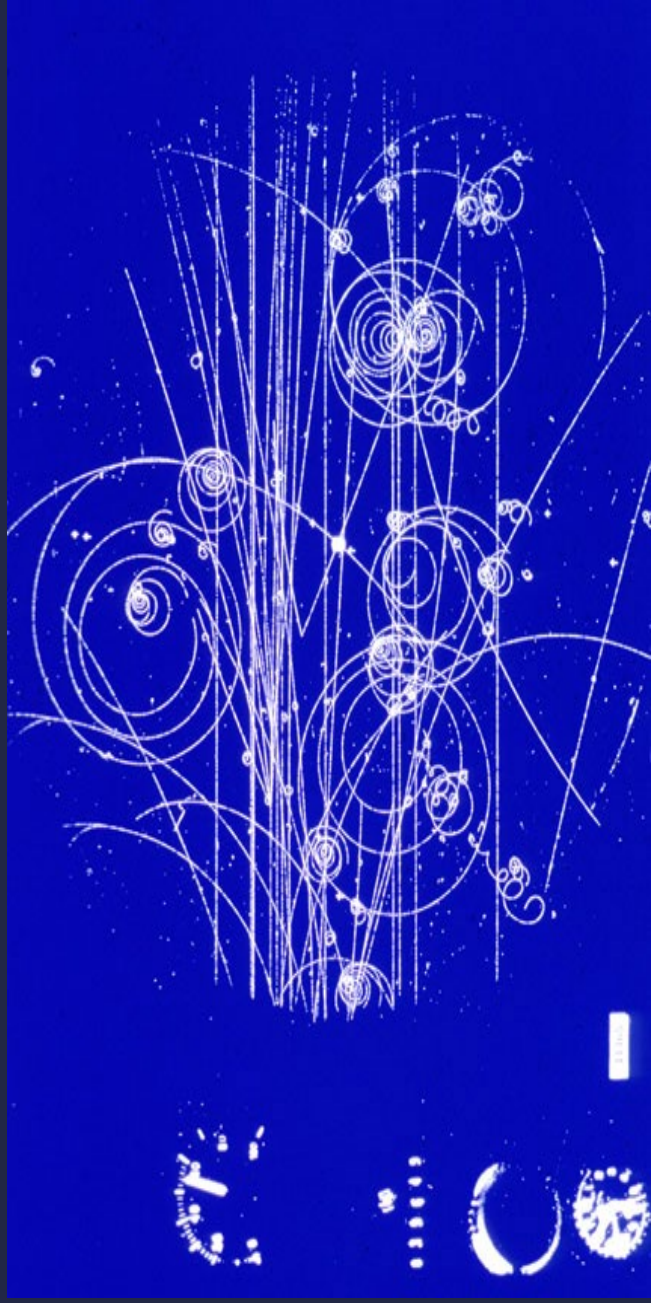
# Geant4 evolutions in parallelization

1. Sequential mode: original since Geant4 v1.0 (1998)
   - Single core (thread) does everything
2. Job level parallelism mode: since Geant4 v9.3 (Dec. 2007)
   - G4MPI for multi MPI implementations
3. Multithreaded event-level parallelism mode: since Geant4 v10.0 (Dec.2013)
4. Task-level parallelism follows



# GPU efforts

- Lots of attempts to try machine translation of Geant4
  - C++ to CUDA
  - None of them were much successful
    - Geant4 has a sophisticated geometry description capability
- MPEXS, a CUDA radiation transport simulator
  - State-of-the-art software developed by the KEK Geant4 team
  - Forked from Geant4
  - I'll be back to MPEXS later



# Brief History of Geant4

Pre-historic

- Early discussions at CHEP 1994 @ San Francisco
  - "Geant steps into the future" R. Brun *et al.*
  - "Object oriented analysis and design of a GEANT-based detector simulator" K. Amako *et al.*

R&D phase (RD44)

Dec '94 – R&D project start

Apr '97 – First alpha release

Jul '98 – First beta release

Production phase

Dec '98 – First Geant4 public release - version 1.0

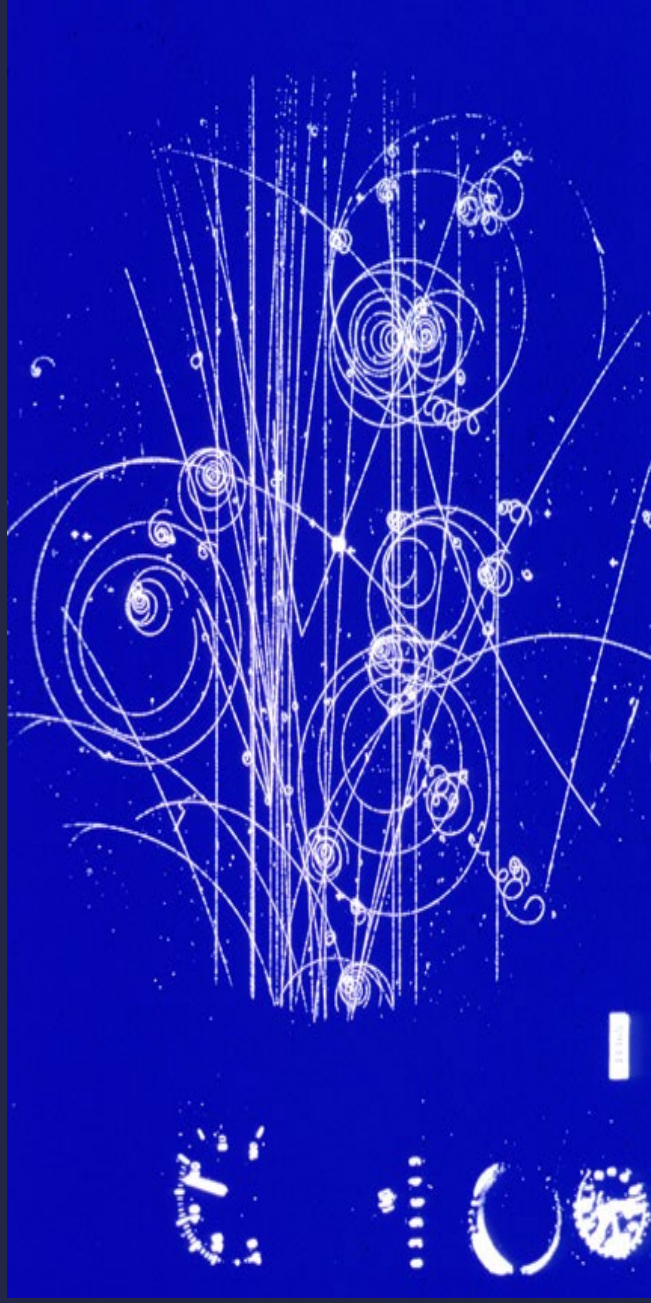
Dec. '13 Geant4 v10.0

- **Several major architectural or design revisions**
- **E.g. STL migration, cuts per region, parallel worlds, multithreading**

Dec 4th, '20 – Geant4 version 10.7 release, task-based parallelism

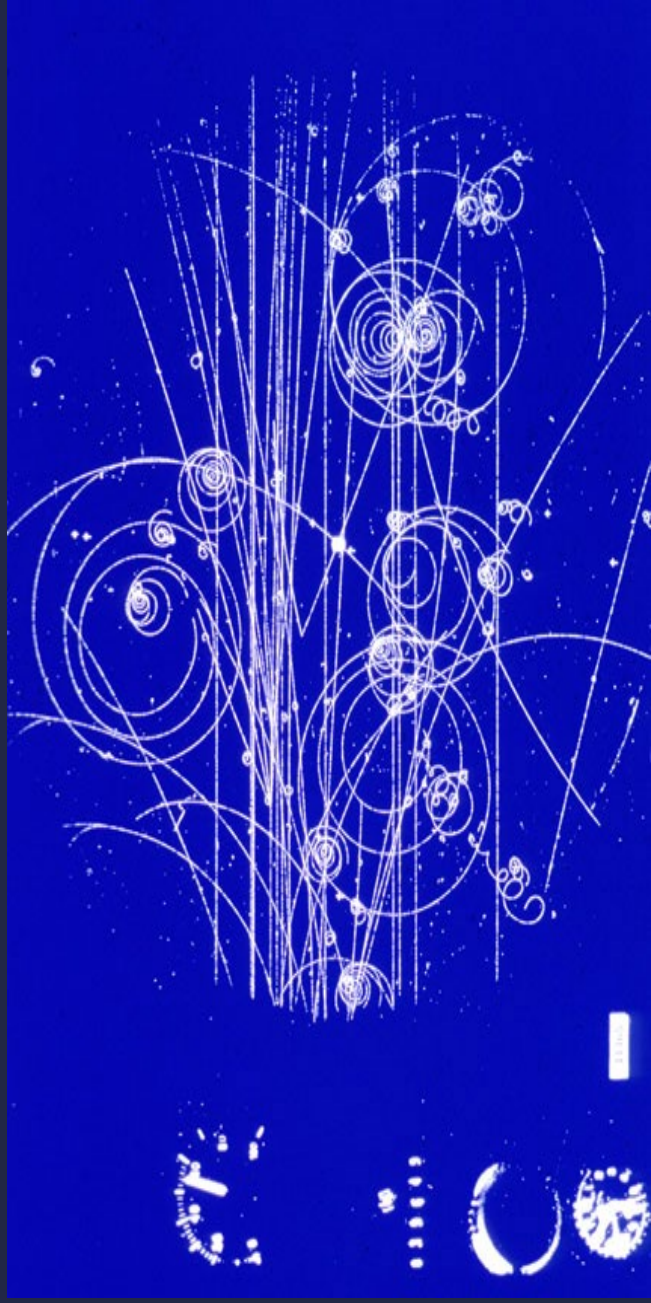
- Nov 19th, '21 – Geant4 10.7-patch03 release

Dec 9th, '22 – Geant4 11.1 release



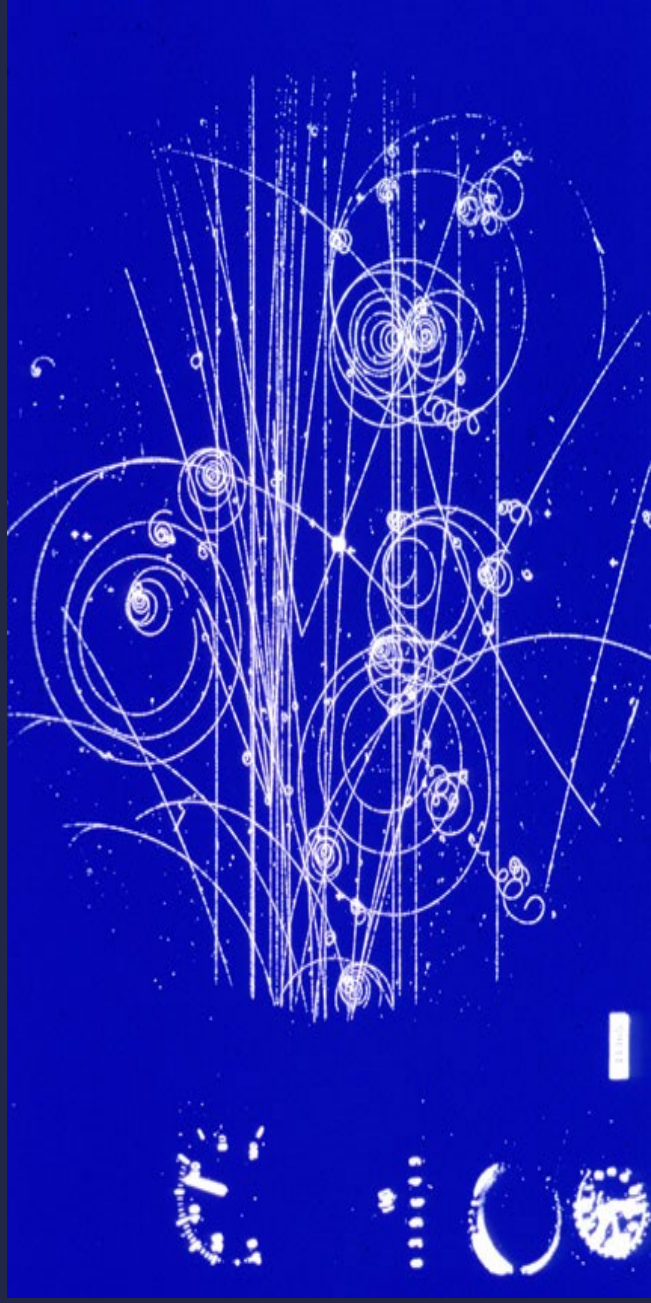
# How Geant4 survives?

- Right decision on design and implementation
  - Strongly related to the History of Geant4
- Collaboration management
  - Leadership in an international collaboration
- Evolution following the technology trend
  - Multi core
- Sustainable development and support
- An International Collaboration
  - Independent from CERN



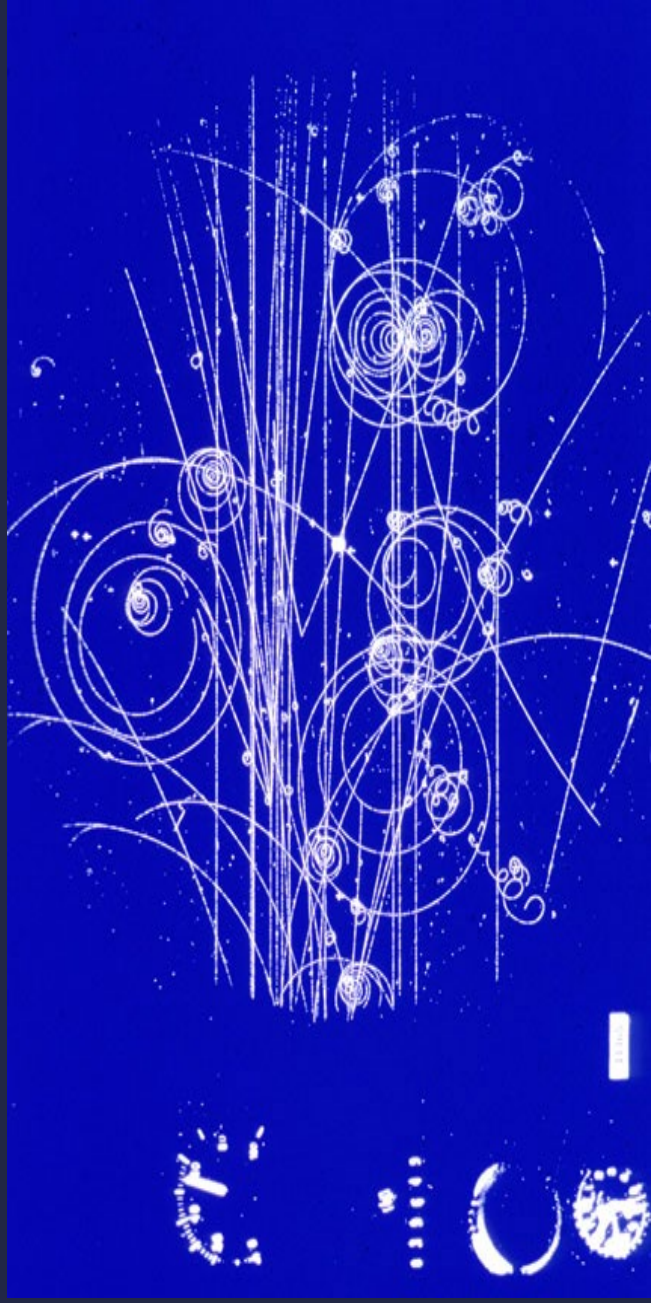
# Background history

- Early 1990's
  - Two large collider projects were ongoing for the Higgs search
    - The Superconducting Super Collider(SSC) @TX, USA
      - Japan participated in the SDC experiment
    - LHC @CERN Geneve, Switzerland
- Oct. 1993
  - SSC was canceled
- 1995
  - Japan joined the ATLAS experiment and contributed for the LHC construction
- 1997
  - the USA joined the LHC experiments and contributed for the LHC construction



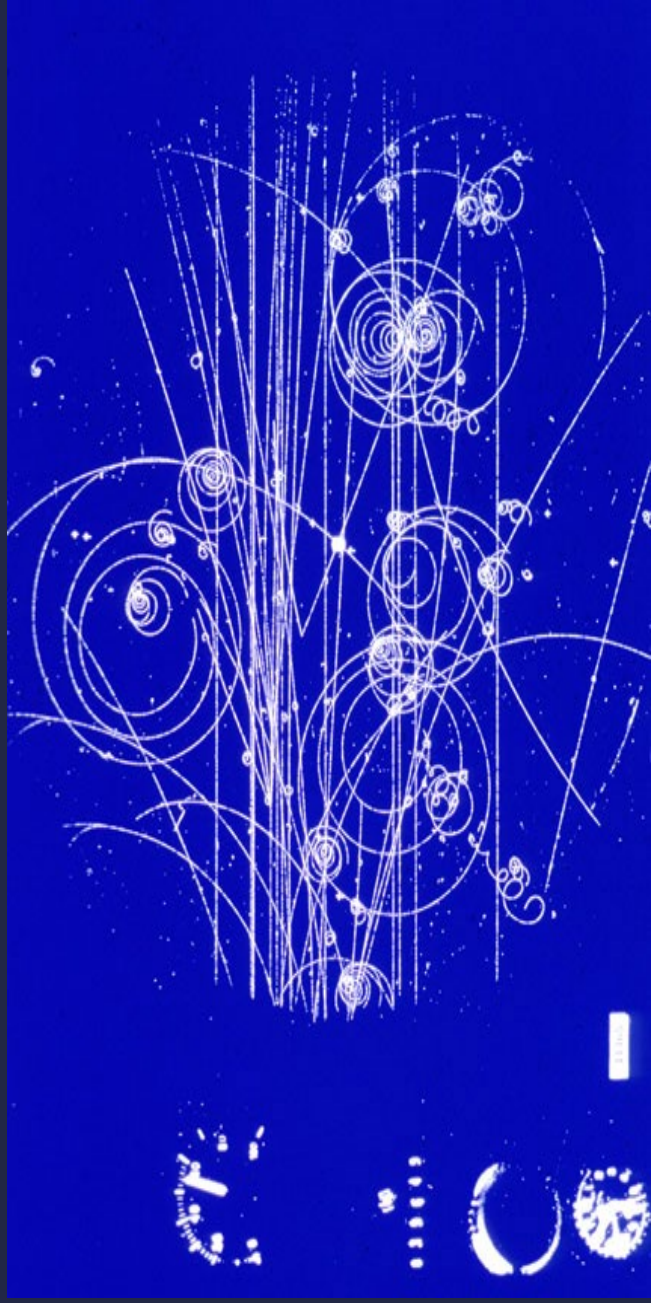
# HEP software in the SSC/LEP era

- CERNLIB was great for a new experiment to start over the software construction, but SSC competed with LHC
- Technically, we were not satisfied enough with CERNLIB
  - User documentations were very well written, but no design document, especially for internal data structures
  - Coding quality was sometimes horrible
    - More than spaghetti
  - Undocumented features often
  - No way to add new functionality by users
- Maintainability was the biggest issue
  - More than 30 years (at least) lifespan of SSC/LHC
  - Accelerator construction took ten years+ twenty years+ operation



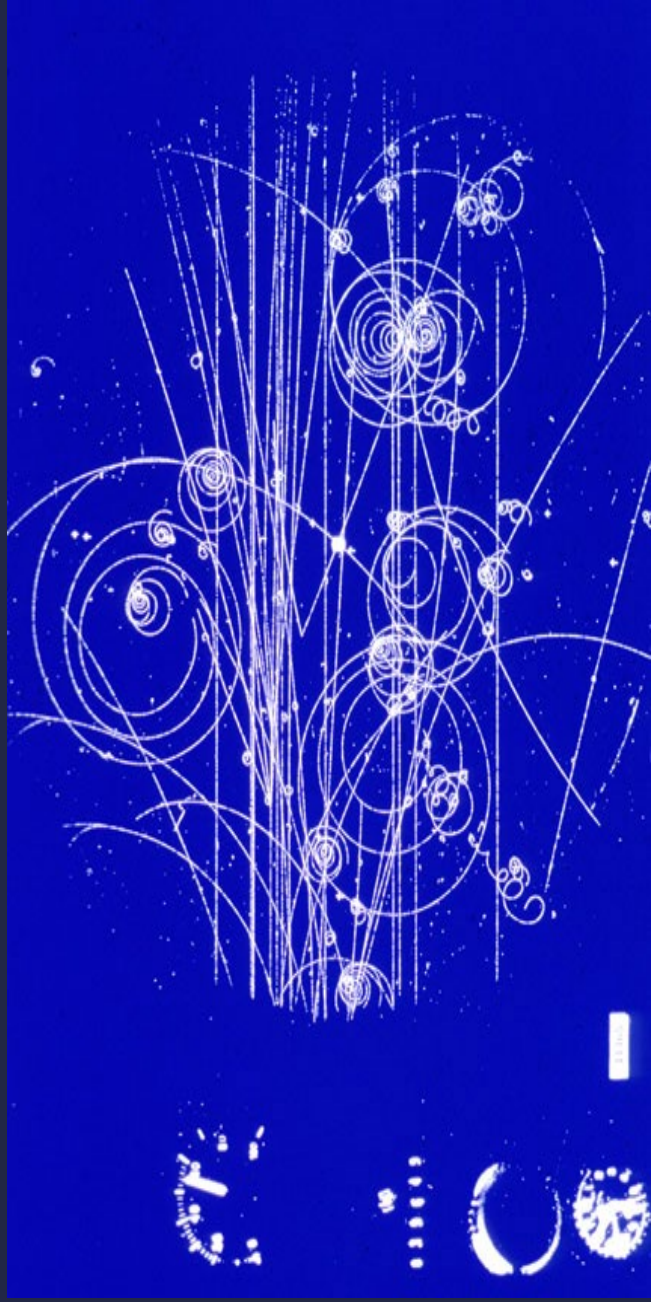
# SDC Japan

- Japanese group decided to contribute to the detector construction and software in SDC
- Started to work on the construction of detector simulation software based on GEANT3
  - Integration of software parts developed independently by geometrically distributed subgroups was more than the nightmare
  - We felt that FORTRAN isn't the language for development and maintenance in the coming three decades
- Decided to develop a replacement for GEANT3 using OOA/OOD/OOP



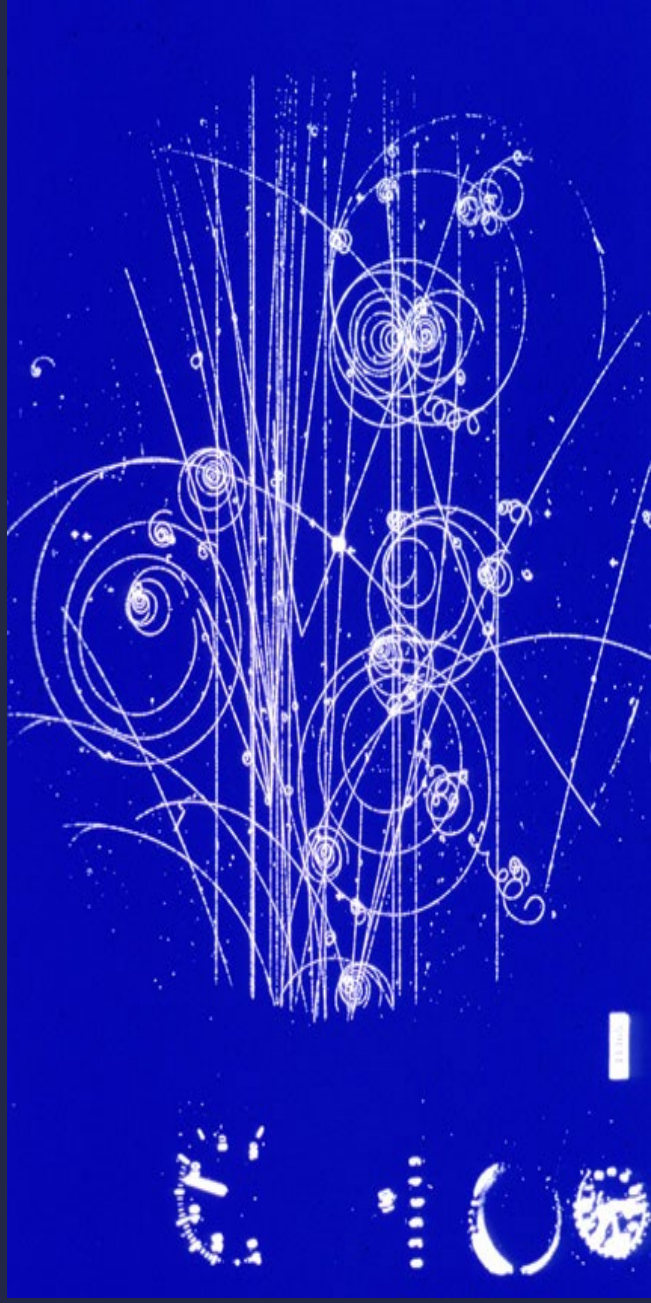
# Software maintainability

- Supposed 30 years of life for the software with 1M lines of code
  - Multi generations of software developers involve further development and maintenance
  - For latecomers, design documents are essential to understand the software quickly
- We sought a systematic and schematic way to record the discussion on the design
  - Software evolves continuously, and the detail of the design changes time by time
- Finally, we encountered the Object-Oriented technologies



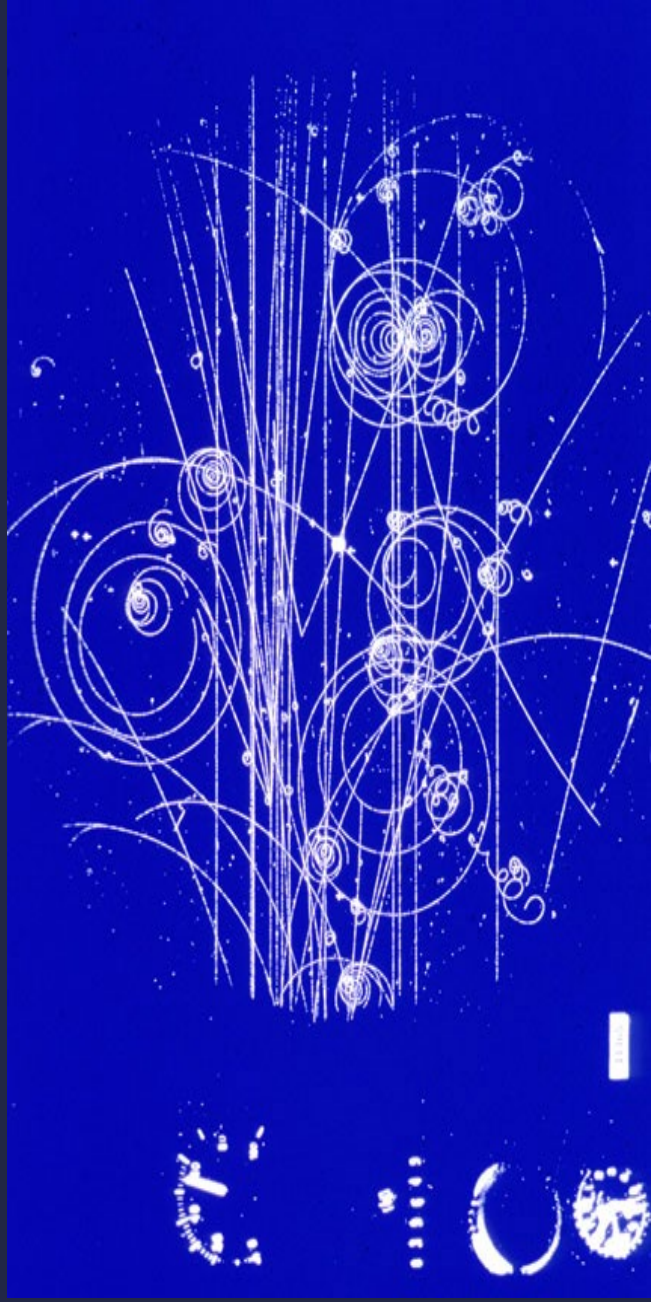
# Rise of OO in early 90'

- OOP already had a long history since 60'
  - C++ has matured enough finally
    - The C++ Programming Language 1990
    - The Annotated C++ Reference Manual in 1990
    - The C++ Programming Language 2nd edition 1991
- OOA/OOD methods finally converged into UML in 1994
  - OMT, OOSE, and Booch method in 1990
  - Standard for the class and other diagram descriptions
- In eraly1990', all the necessary tools for OO had come
  - Major computer vendors started to sell C++ compilers
  - Many people started to develop their software with those new tools



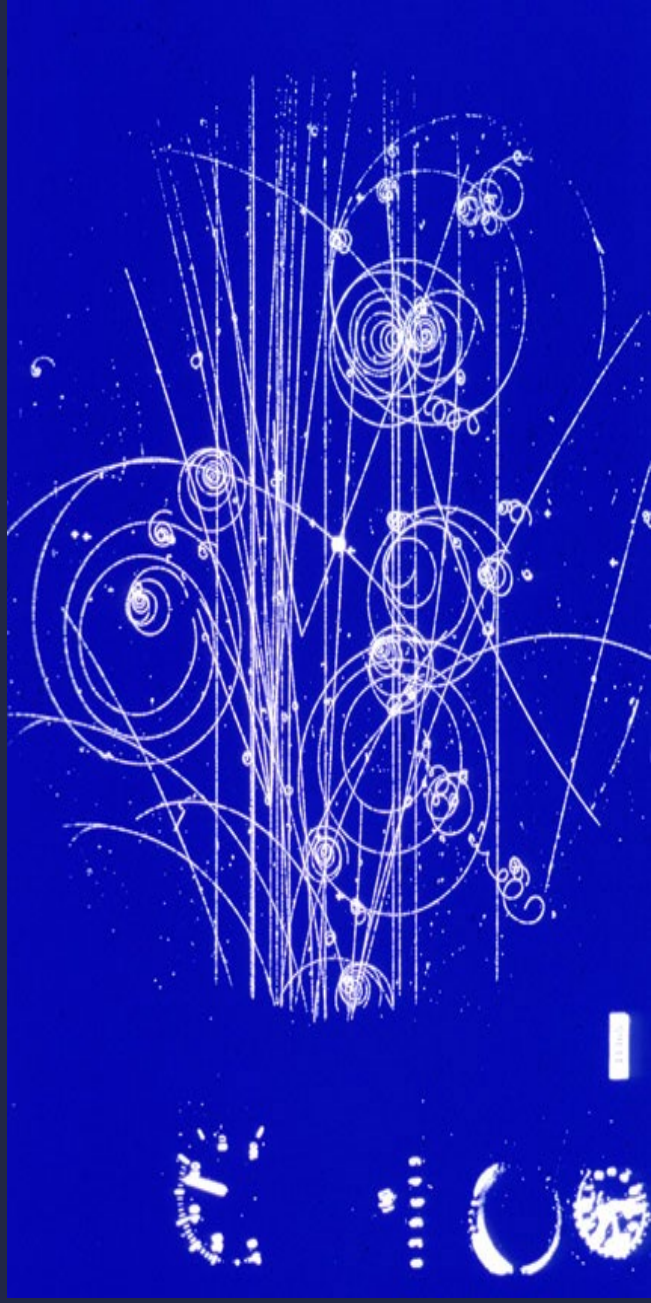
# Japanese prototype

- "ProdiG: Toward Object Oriented GEANT", Y.Takaiwa, 1993, MC93
  - Started in 1992
  - Drawn class diagrams using OMT (James E. Rumbaugh)
- Big discussion between reengineering GEANT3 or starting from the scratch
  - Discussion went long and looked continued forever
  - We decided to go reengineering GEANT3
    - HEP jargon, such as RUN, EVNET, TRACK, etc., are brought into our class diagram at this stage
    - We didn't know the detailed usage of radiation simulators in other fields



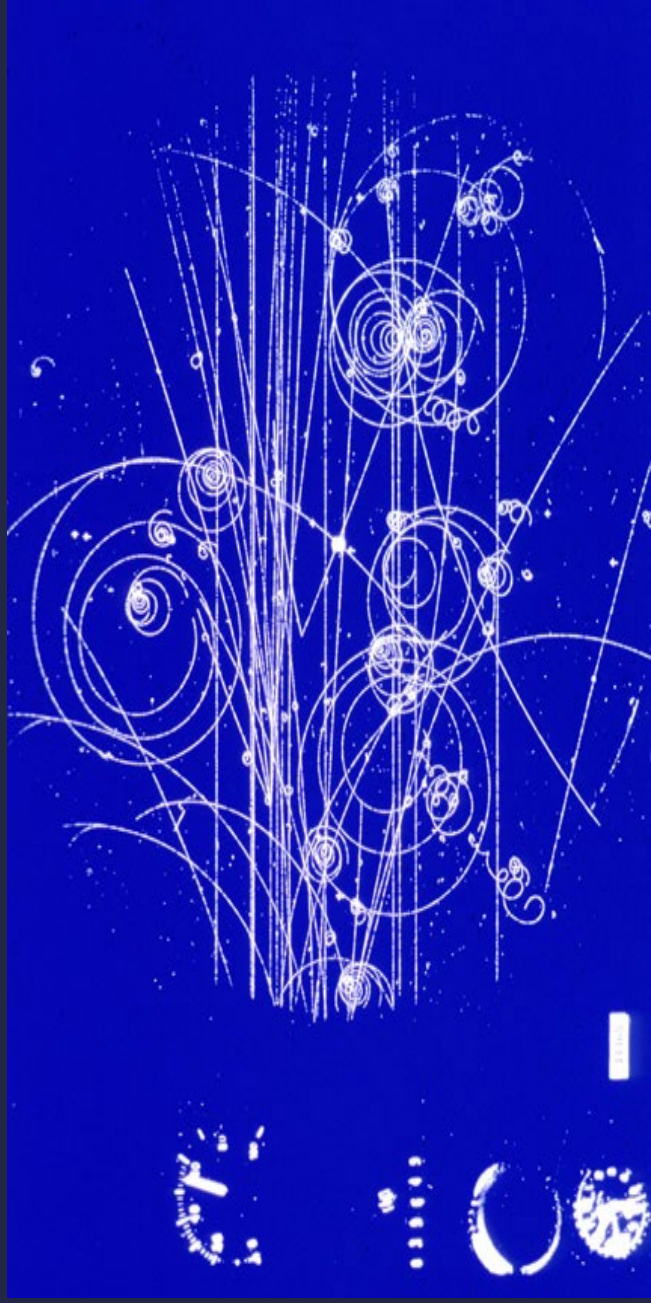
# CERN GEANT team prototype

- Simone Giani and Paul Kent started the development of a C++ class library for detector geometry description in 1992
  - Paul Kent was a graduate student working as a summer student at CERN
  - This project was secret because nobody wanted to use another programming language other than FORTRAN at CERN at that time
    - The FORTRAN giant, Michael Metcalf, had his office the next door
- Rune Brun was thinking of developing an alternative for CERNLIB toward LHC



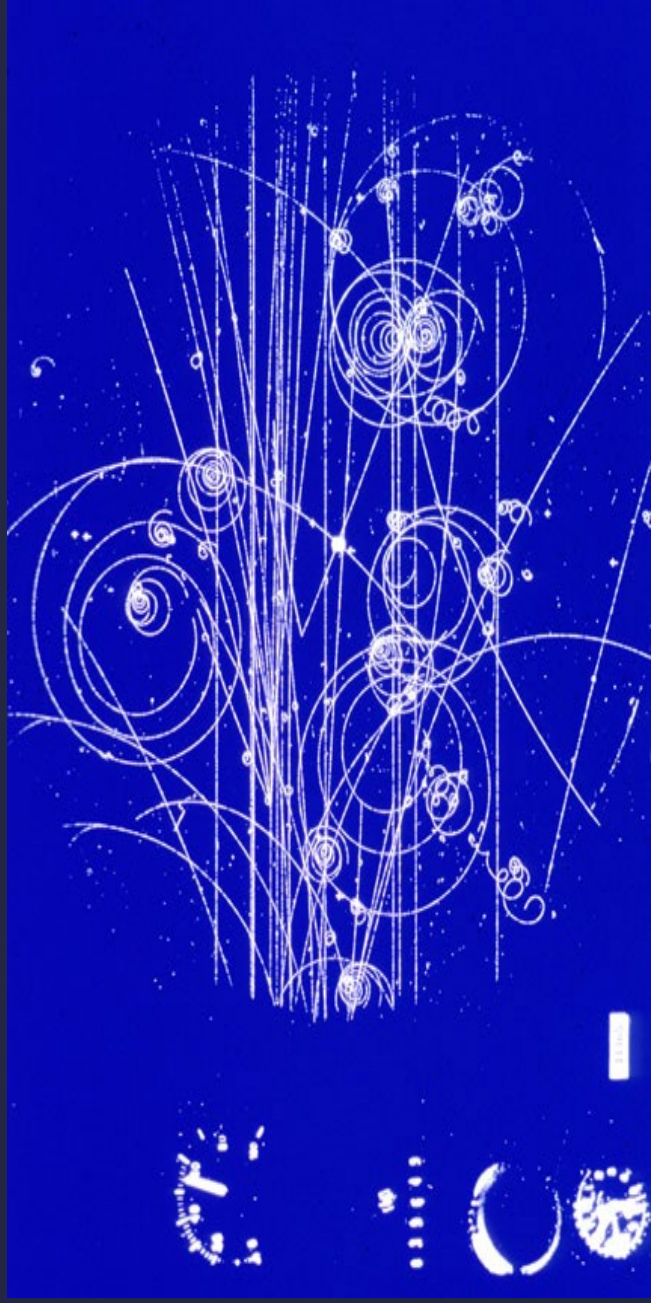
# GEANT3

- A part of CERNLIB
- The first radiation simulation software with detailed geometry description functionality with the complete set of physics processes
  - GEANT= GEometry ANd Tracking
  - Integrated graphics
- Written in FORTRAN
- Rune Brun led the development at that time
  - Project lead and principal designer



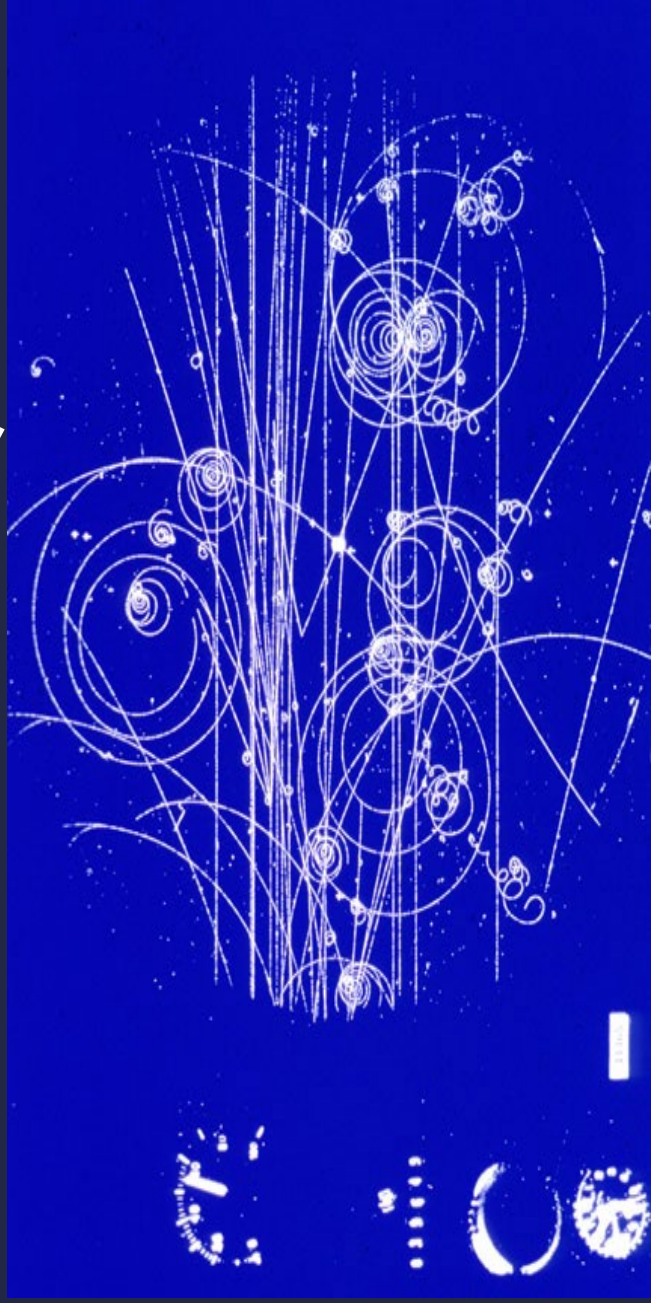
# CERNLIB

- PATCHY was used for code management
  - FORTRAN had no header file
- All functionalities necessary for a HEP experiment were covered
  - PATCHY for code management
  - ZBOOK/ZEBRA for data management
  - GEANT3 for detector simulation
  - HBOOK for histogramming
  - PAW for visualization
  - HIGZ for graphics
  - MINUIT and other packages for numerical calculation
  - User contributions



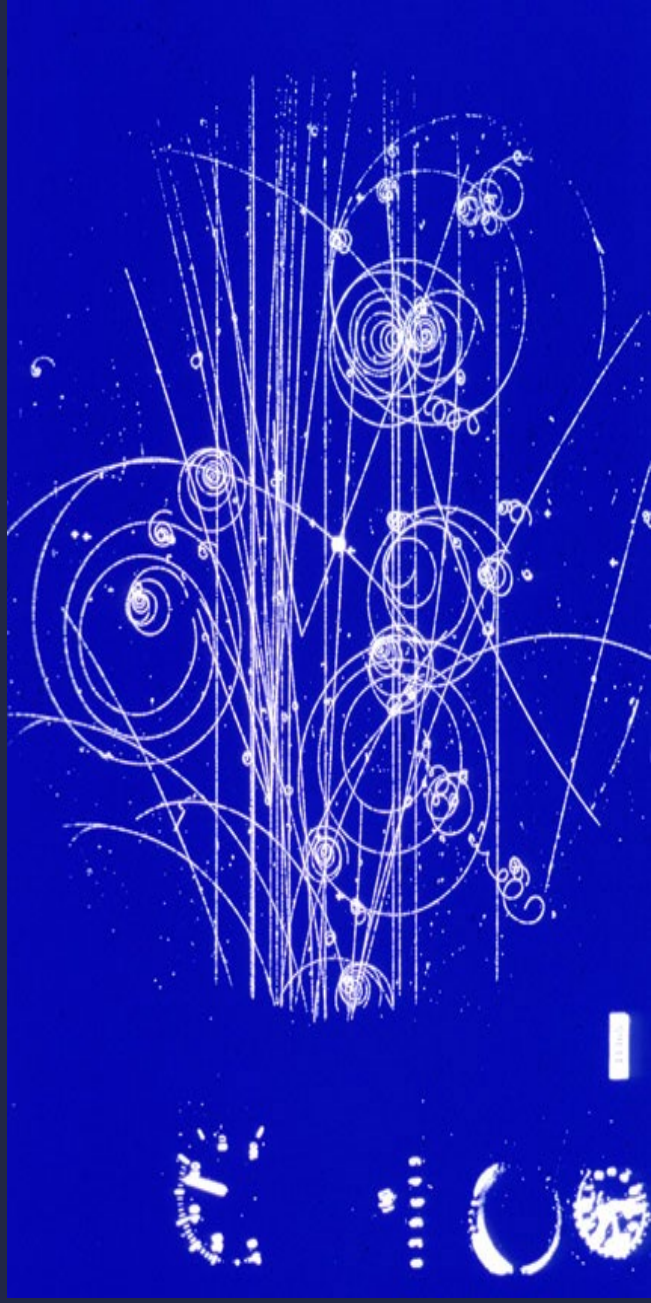
# RD44 (CERN R&D)

- CERN GEANT and Japanese groups met at MC93, mini-workshop at CERN in 1993 and CHEP94 for discussing about the future of GEANT3
  - CHEP94 was held after SSC was terminated
- Both groups agreed to collaborate and make a proposal to CERN for research funding as P58
  - Approved as RD44 (1994-1998)
    - Many other people from lots of institutions joined us
  - Led by Simone Giani
    - Rune Brun had left CERN IT
- After the first version of Geant4 was released in 1998, the Genat4 collaboration was established
  - 100 developers from more than ten countries



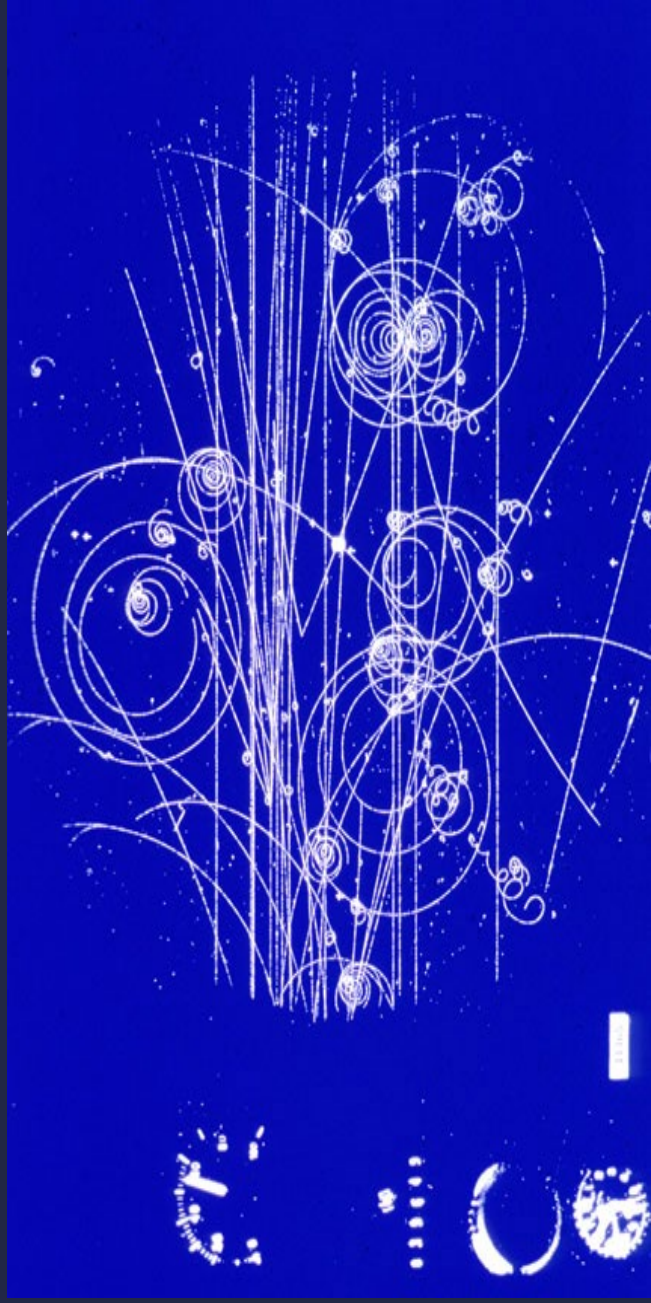
# Basic requirements on Geant4 in HEP

- 30 years of life
  - Maintainability
    - Following the evolution of the computing environment
  - Extendibility
    - Users can add any new functionality easily by themselves
  - Active developments over the life
    - Lower threshold for latecomers
      - Design documents
- Capability to simulate the response of the detectors at LHC



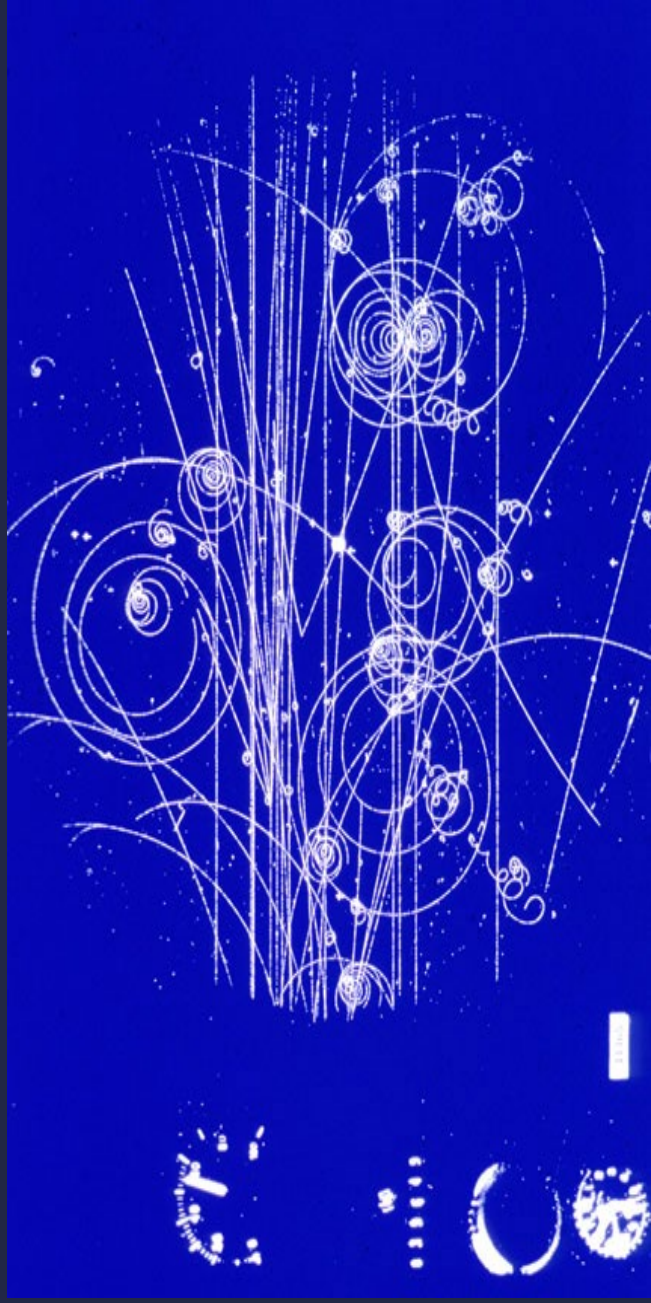
# Wider requirement sampling during RD44

- Not only HEP but also medicine and space were under our target since the beginning
  - ESA (European Space Agency) joined us
  - Interviewed medical physicists
    - Particle therapy (proton and carbon)
- Learned other software for radiation transport simulation not only GEANT3, but also, FLUKA, MCNP, EGS, and so forth.



# What we agreed at the beginning of RD44

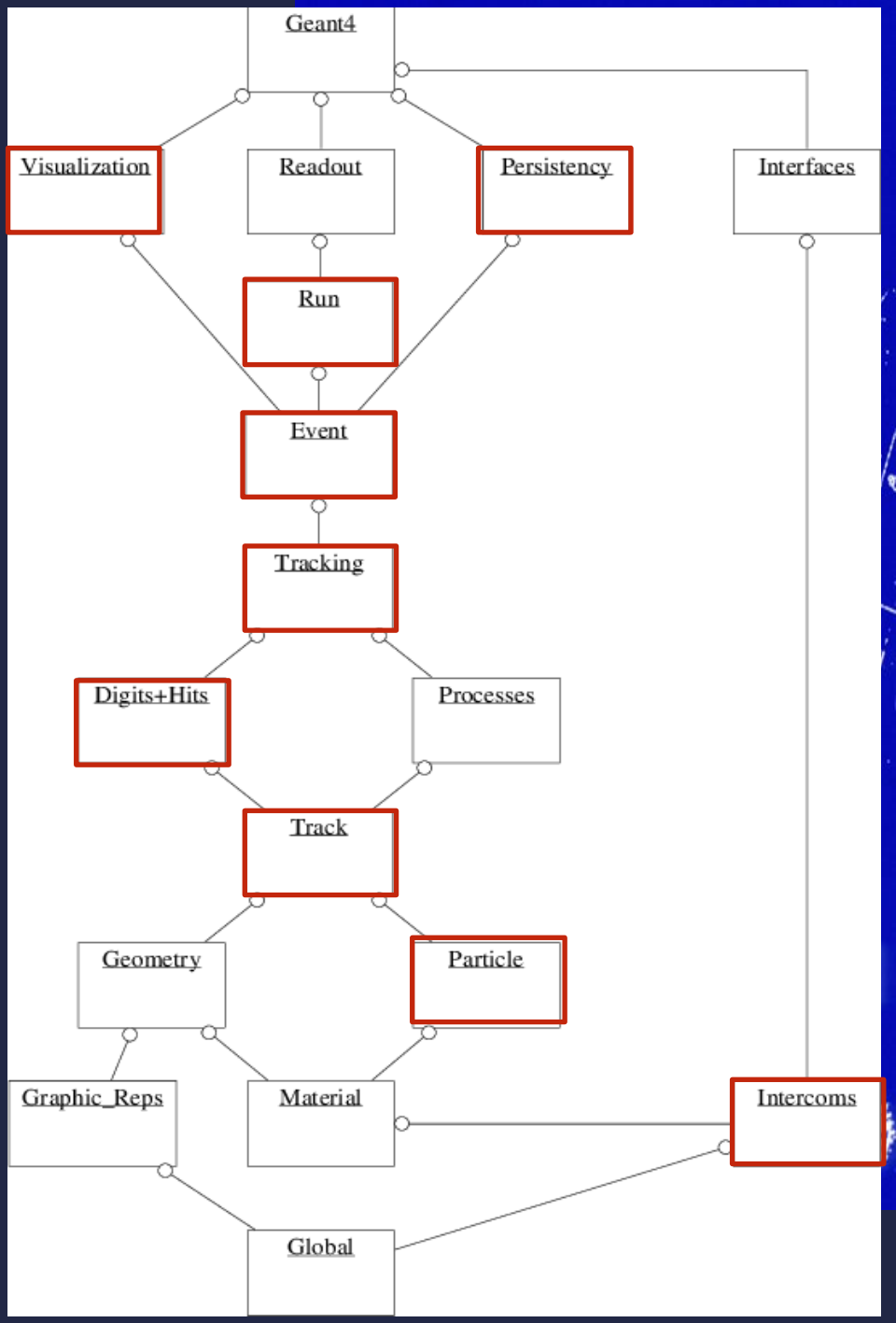
- The product name is "Geant4" not "GEANT version 4"
  - Should be "Geant4" not "GEANT4"
    - We wanted to reveal that we are not using FORTRAN
- Toolkit approach
  - No main program
- All diagrams and documents should be maintained and kept very well
  - We first deployed OMT, then the Booch method and UML followed
- We have "Class categories"
  - Originally in the Booch method, but not in UML
    - A group of classes
    - We continued to use this concept with UML
  - One working group for one class category



# Geant4 class category diagram

- Each class category has a working group
- Each working group has independent governance and development goals
- Steering Board decides the development goals for each year

Major Japanese contributions:
Visualization, Run, Event, Tracking, Digits+Hits, Track, Particle, Interfacoms and Persistency

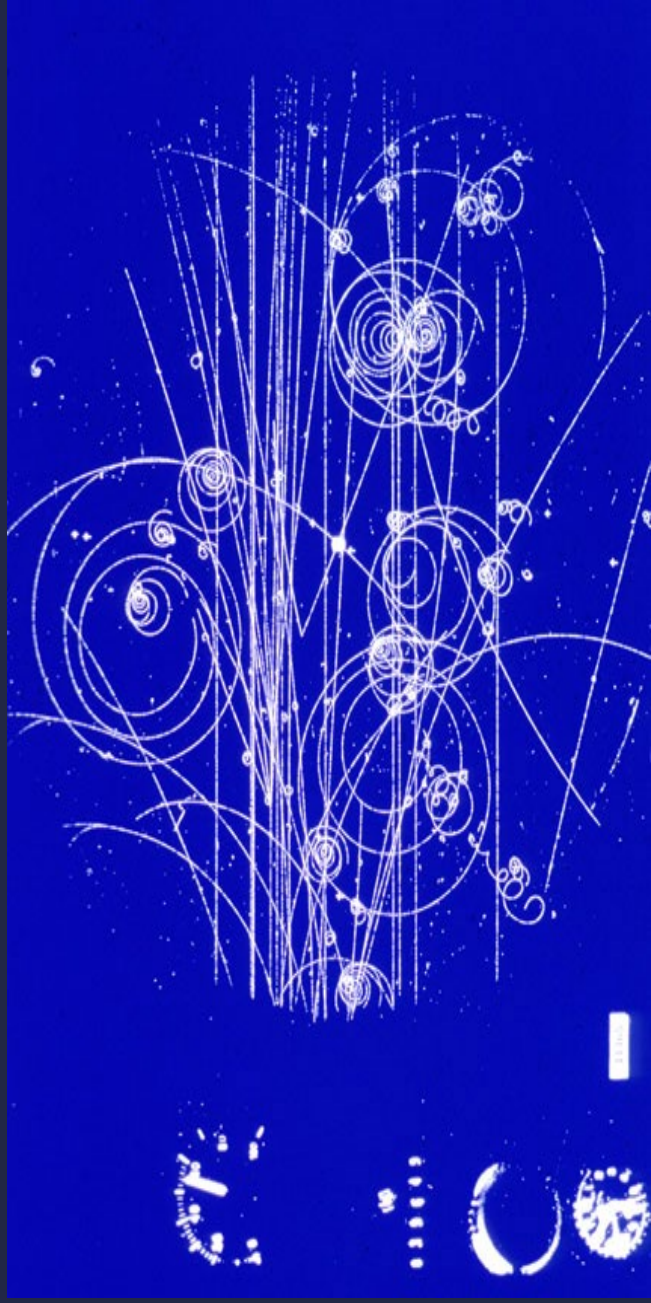




Takashi.Sasaki@kek.jp

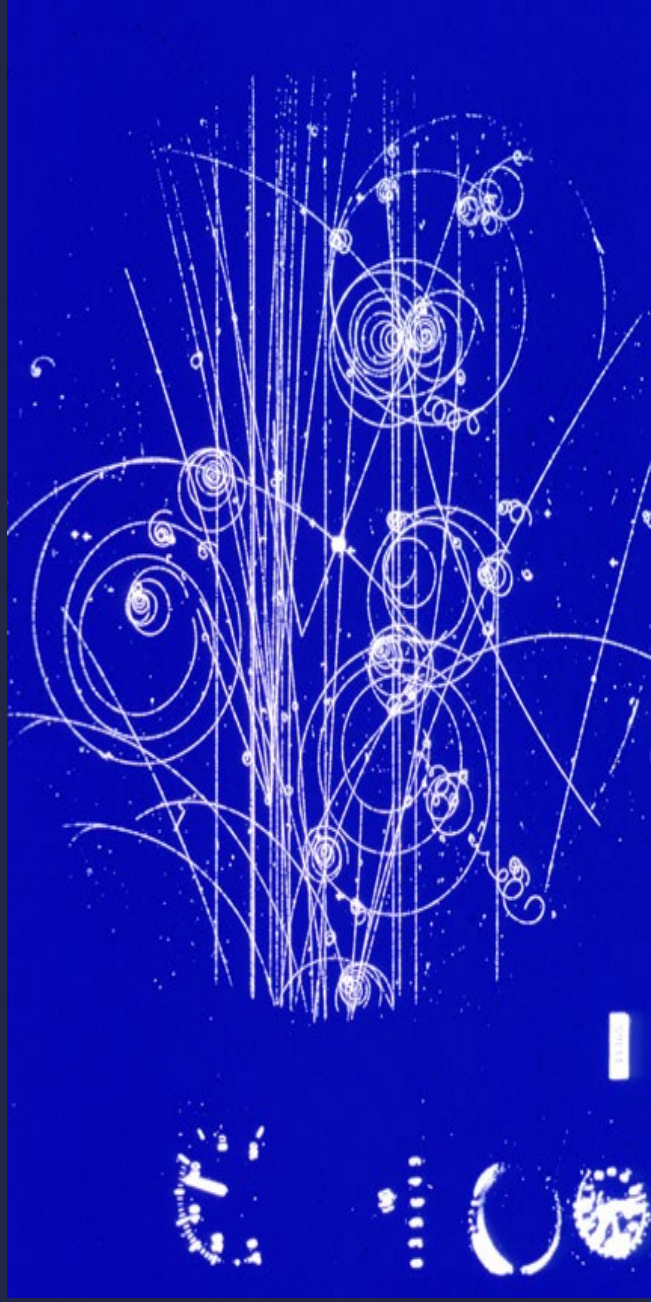
# Centralized to distributed software development

- CERN had an extensive software development team
  - Still, they need more manpower for LHC software
- New age had come with the Internet; World-scatted small groups may work in parallel and independently in software development
  - World Wide Web for information sharing
    - WWW was born in 1990
  - Code repository and management systems
    - AFS, CVS, then git



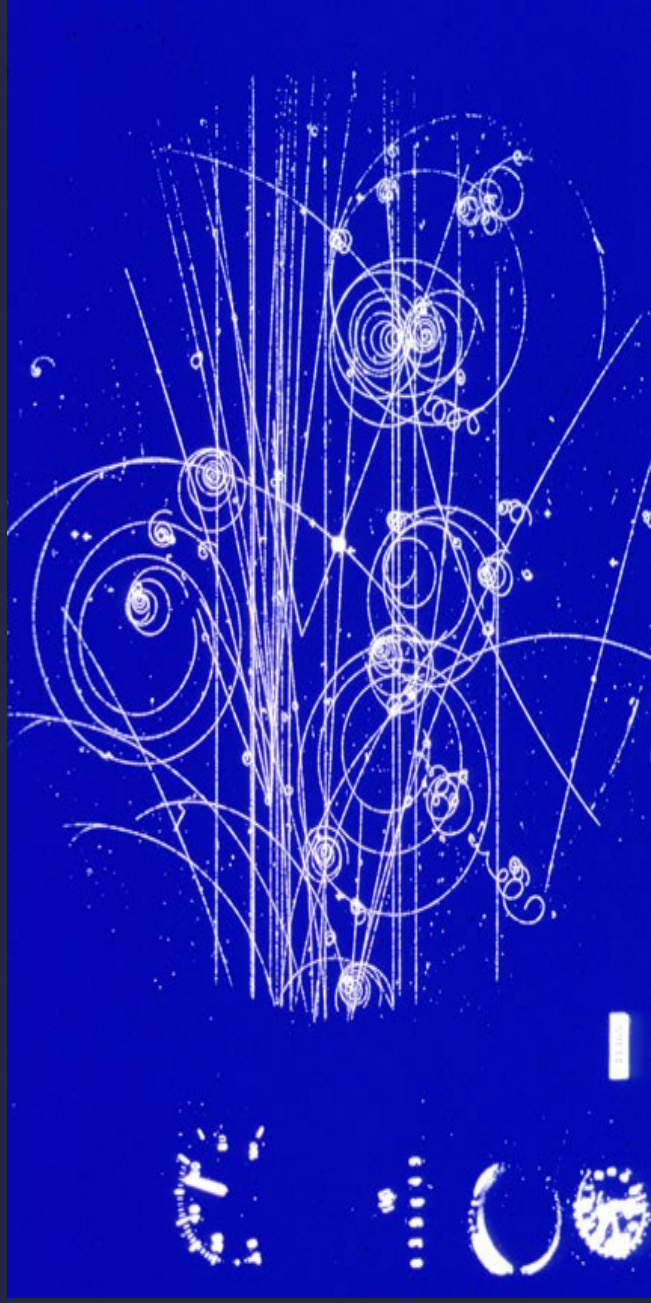
# OOA/OOD/OOP

- For the first two years, we spent time on the design discussion
  - Based on earlier Japanese design, redraw lots of diagrams
  - Japanese people knew such discussions never be quickly converged from the previous experience
- Prototype development ensure the power of OO
  - Integration of independently developed software parts went much easier
  - We confirmed that we were toward the right direction



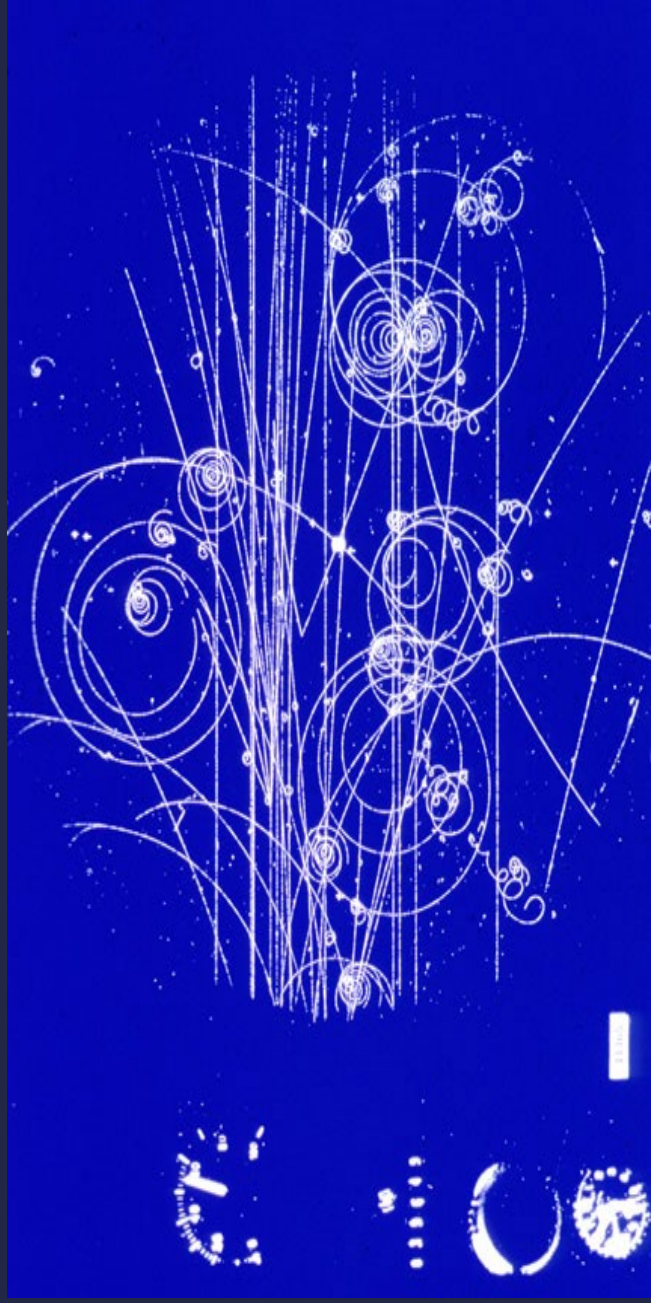
# Responses to RD44

- FORTRAN lovers upset a lot
  - Lots of FORTRAN lovers were around CERN
    - CERN IT was contributing to the standardization of FORTRAN
  - They said C++ is not a program language for physicists
- Since Fortran 2003, Fortran became an object oriented language
  - If it were 10 years earlier, HEP people could have stayed with Fortran



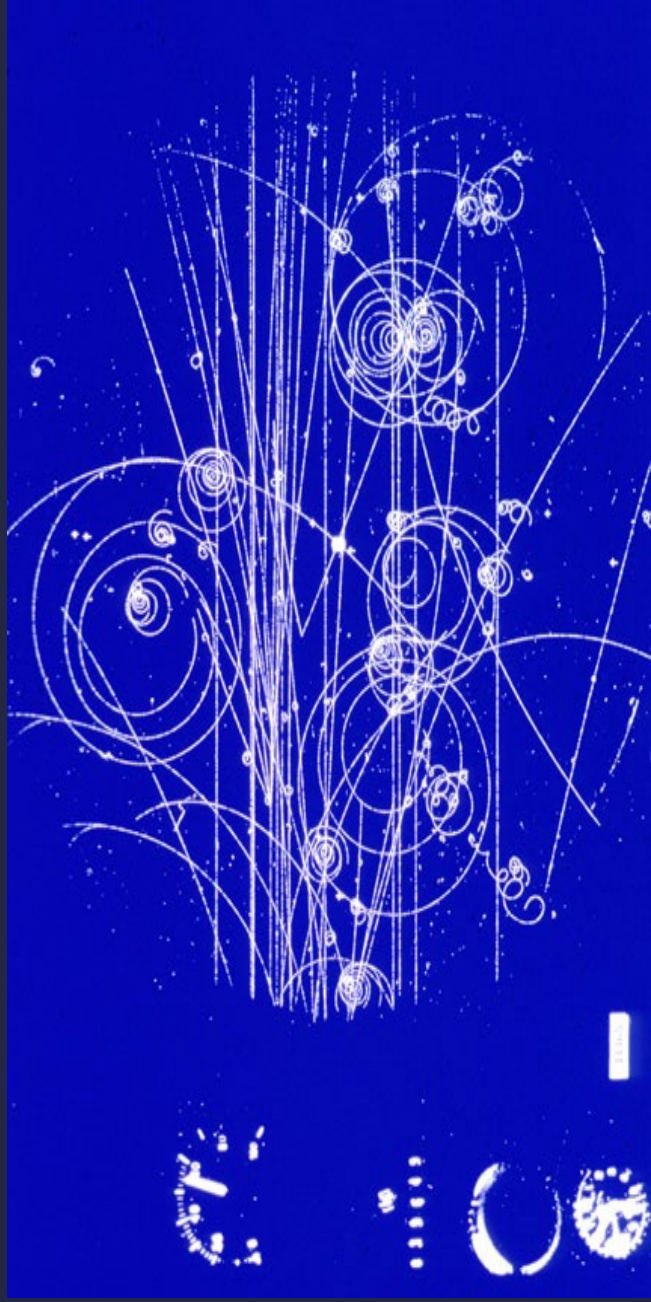
# Geant4 1.0

- The first version of Geant4 released public in 1998
  - Toward the release, the developers had a workshop in Niigata Japan
    - We developed and debugged the code
    - Real tough work
  - Here is another Japanese contribution
    - Hitachi Ltd. sponsored the workshop and rend lots of UNIX workstations during the period
    - Without their help, we couldn't release Geant4 timely
- We opened source code without any charges



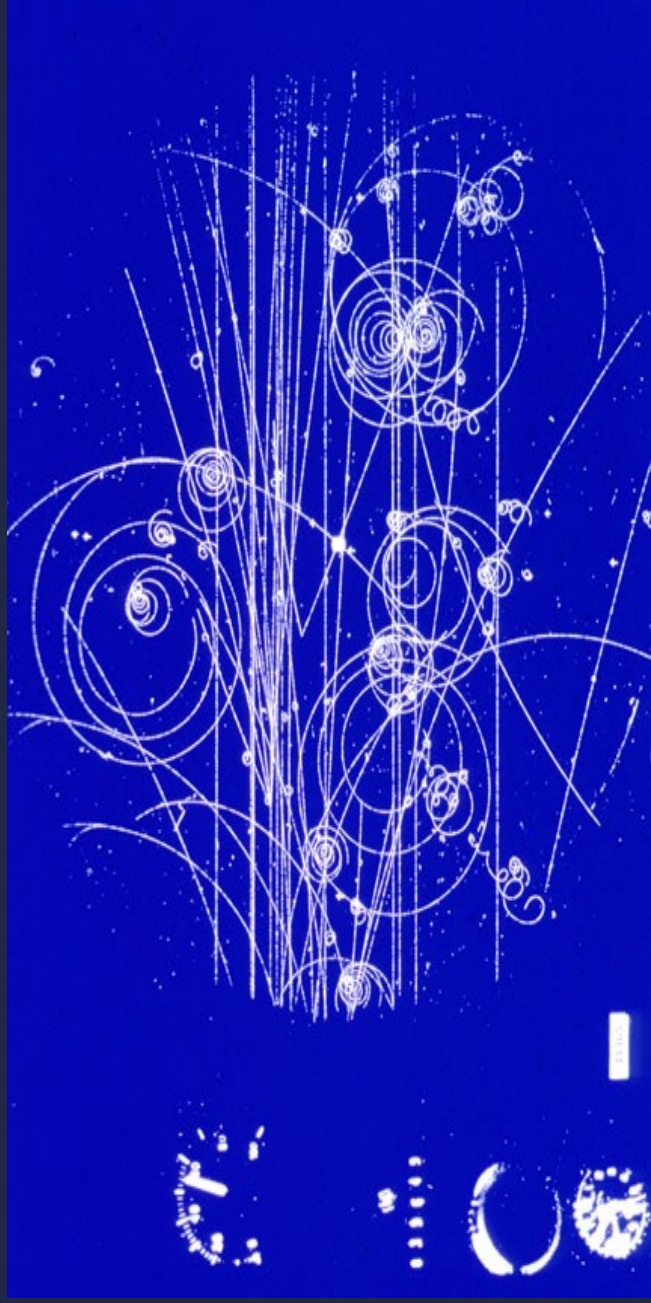
# Documents

- For Users
  - Introduction
  - Installation guide
  - Physics manuals
- For Application Developers
  - How to develop an application using Geant4
- For Toolkit Developers
  - for those who want to contribute to the extension of the functionality to the Geant4 toolkit



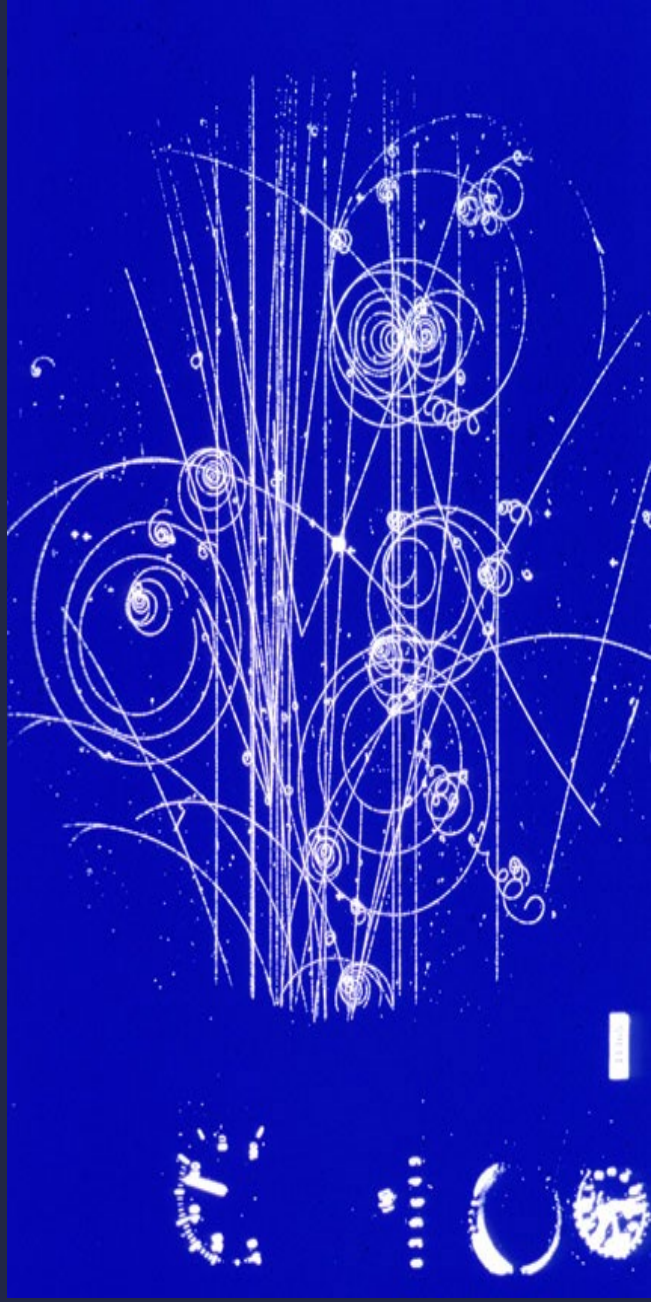
# Examples

- Lots of examples for the different level users
  - Basic Examples
  - Extended Examples
  - Advanced Examples
- New users may try to look the source code and try to run



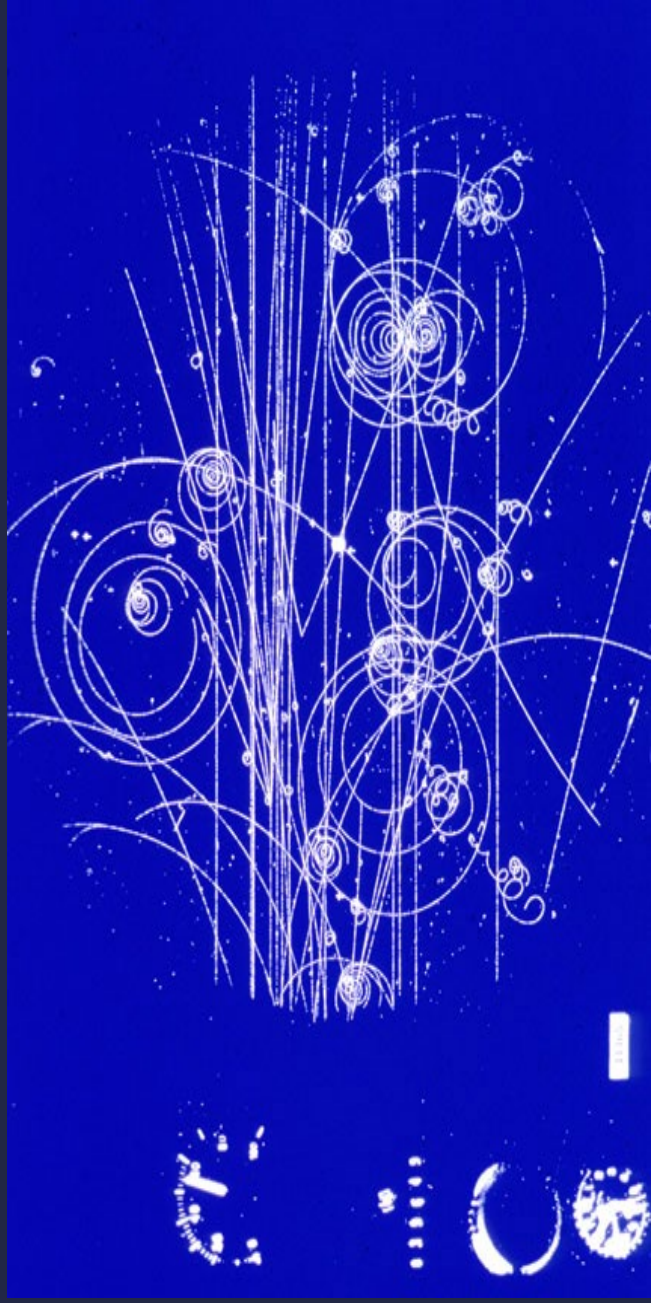
# Training opportunities

- Training courses held regularly everywhere in the world
  - Developers conduct training courses often, but also users organize the training
- Those opportunities were announced on the Geant4 collaboration web page
  - https://www.geant4.org/



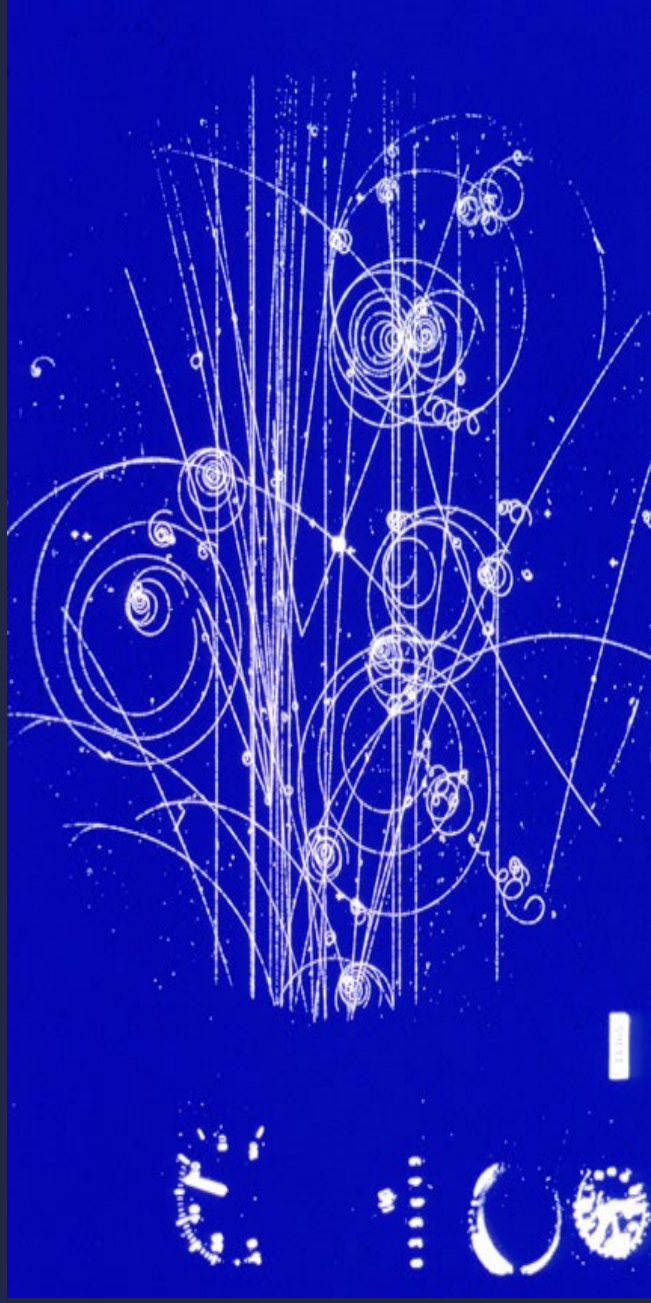
# The Genat4 collaboration

- The collaboration is based on a Collaboration Agreement (CA) among the participating laboratories, experiments and national institutes
- Many specialized working groups are responsible for the various domains of the toolkit



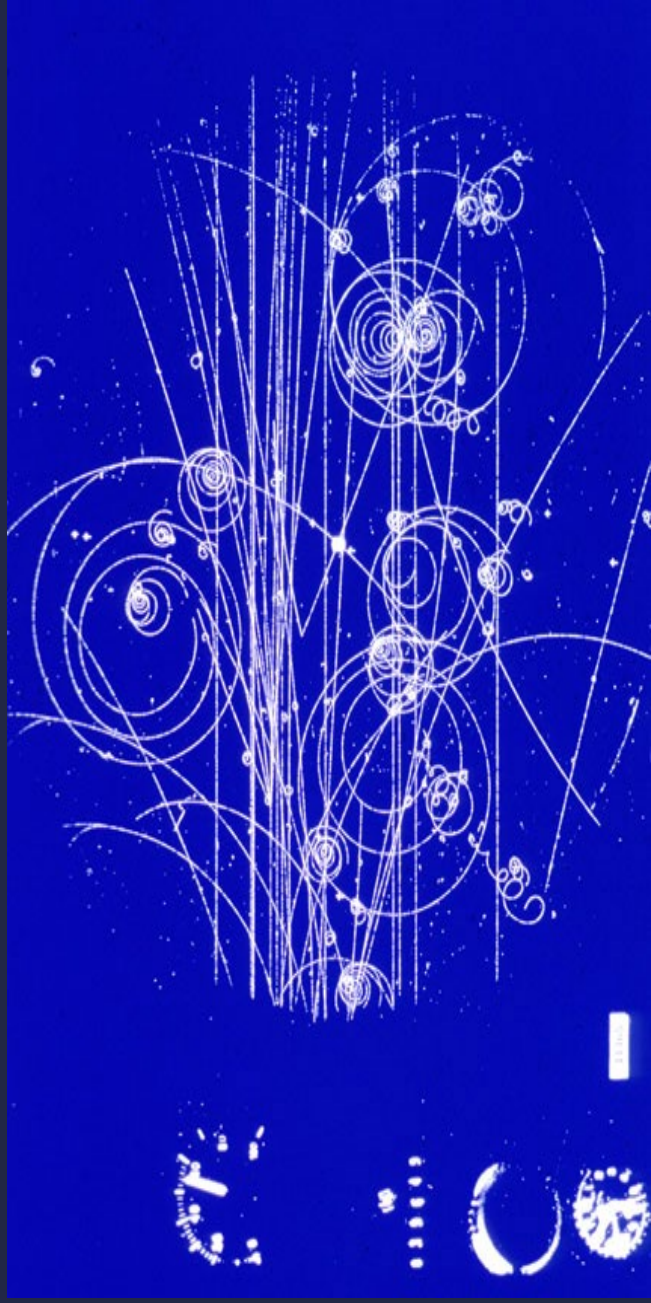
# Collaboration governance

- Oversight Board
  - Representatives of the CA signing body
  - Review the report from Steering Board
- Steering Board
  - Election of the spokesperson
  - All the decisions in the collaboration
  - Approvements of the working group activities
- Working groups
  - Send representatives to the Steering Board
  - Independent governance



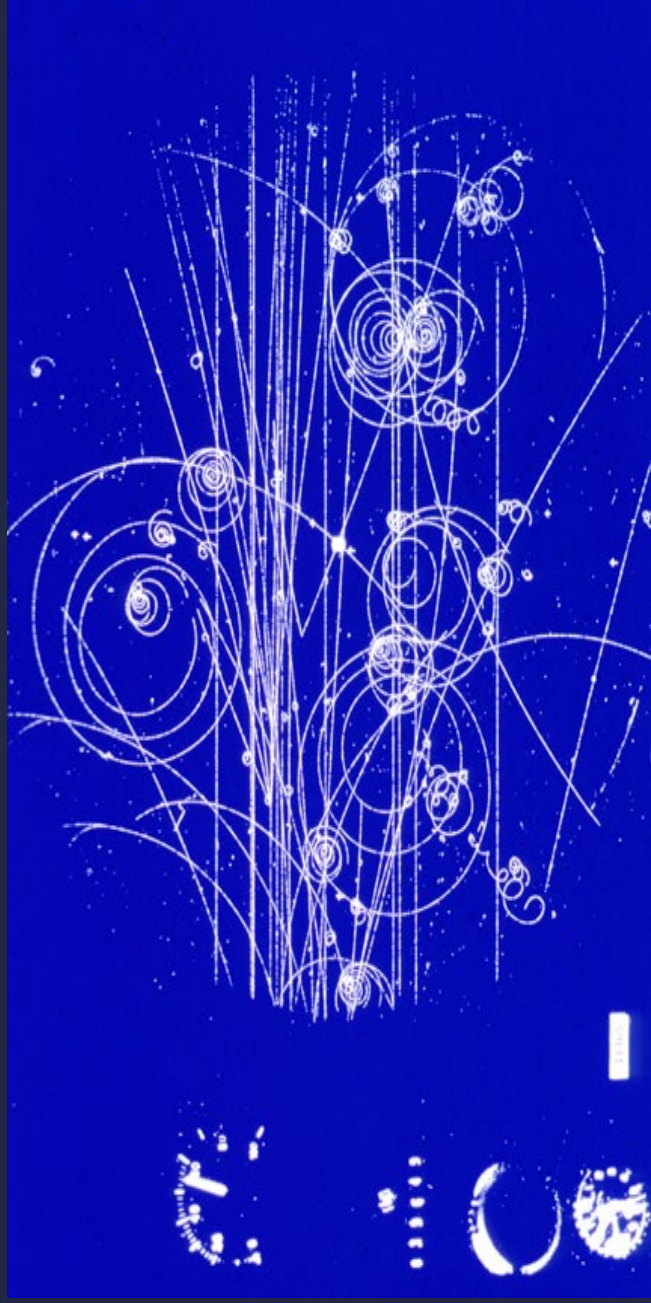
# Spread of Geant4

- The first used experiment was BaBar at SLAC
- All LHC experiments, except one experiment, deployed Geant4 for their detector simulation soon
  - The last experiment became a user of Geant4 almost ten years later
- All ongoing HEP experiments are Geant4 users
- ESA started to use Geant4 for their space missions soon
- We see users in medicine more and more every year



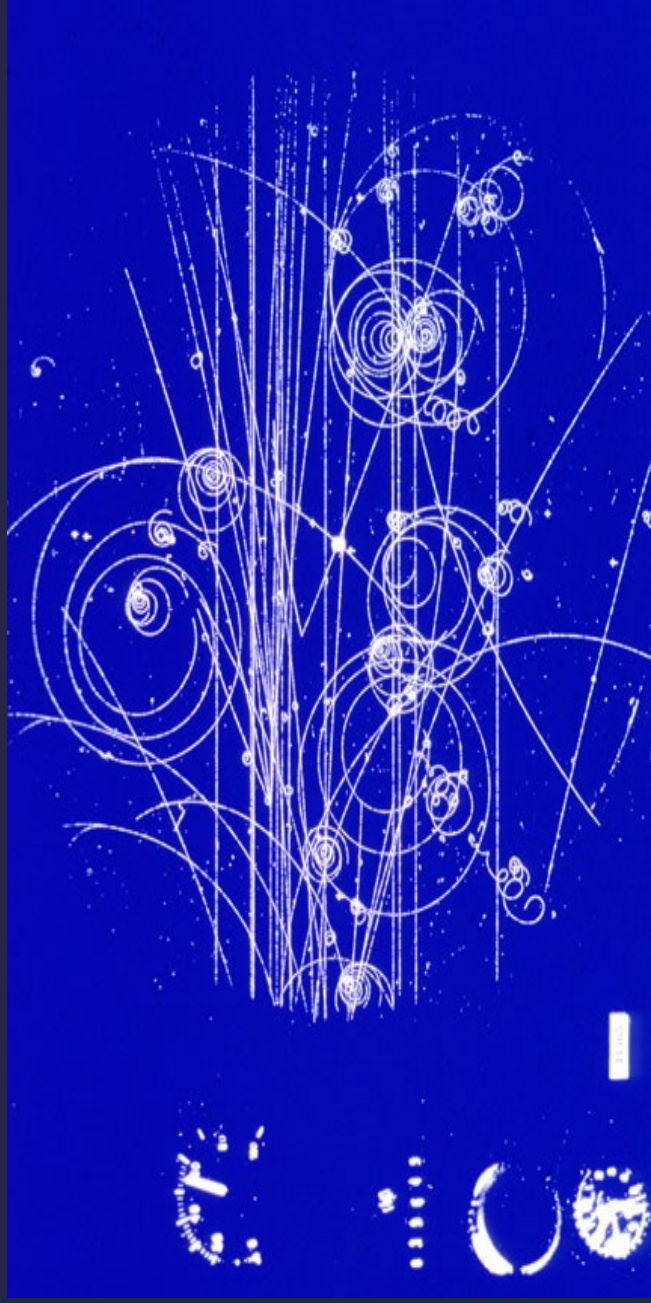
# Commercial users

- They like open source but not GPLed
  - Geant4 License is more like BSD's one
- We identified lots of commercial users at the events, but there are more unknown users
  - Some of them tried to ask a question on our user forum anonymously using a free e-mail address
    - We could imagine their business from their questions
- Some developers from commercial companies
  - Contribution to the development



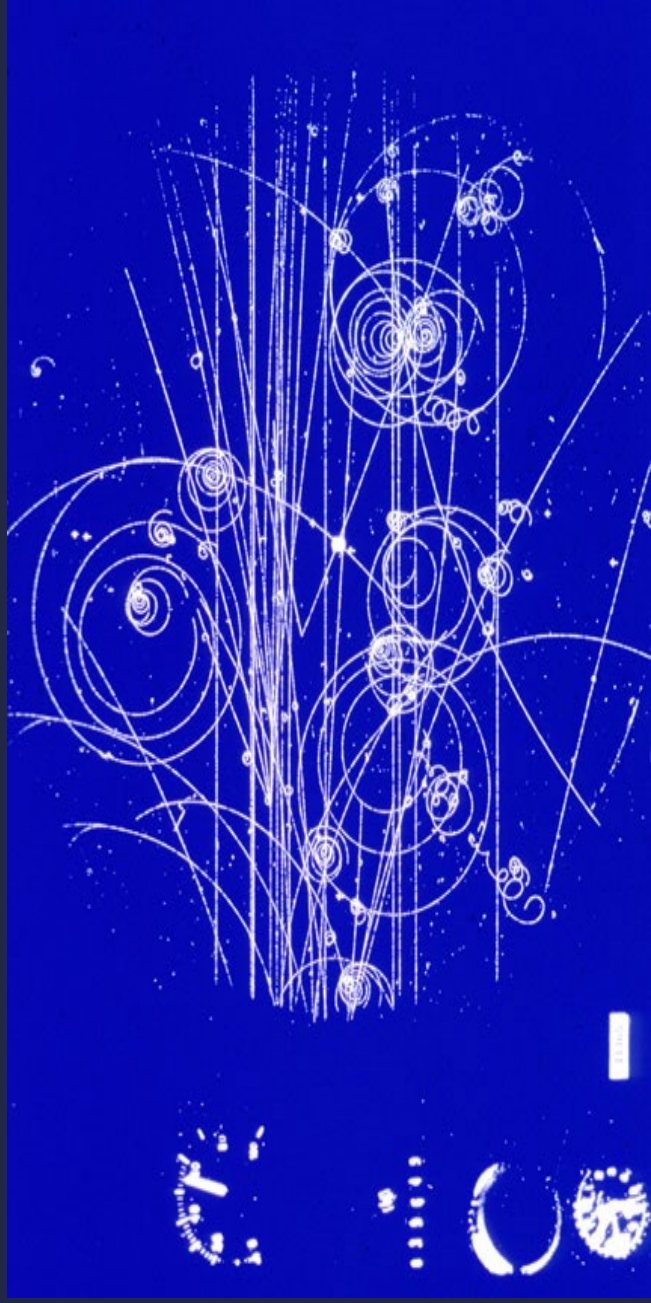
# Great success

- Geant4 is accepted very widely because
  - Open source and free of charges
  - Rich functionality
  - Rich physics
  - Documented very well, and many examples
  - Training opportunities
  - No import/export control
    - Geant4 cannot simulate atomic bombs
- Maintenance and development continued actively over almost 30 years



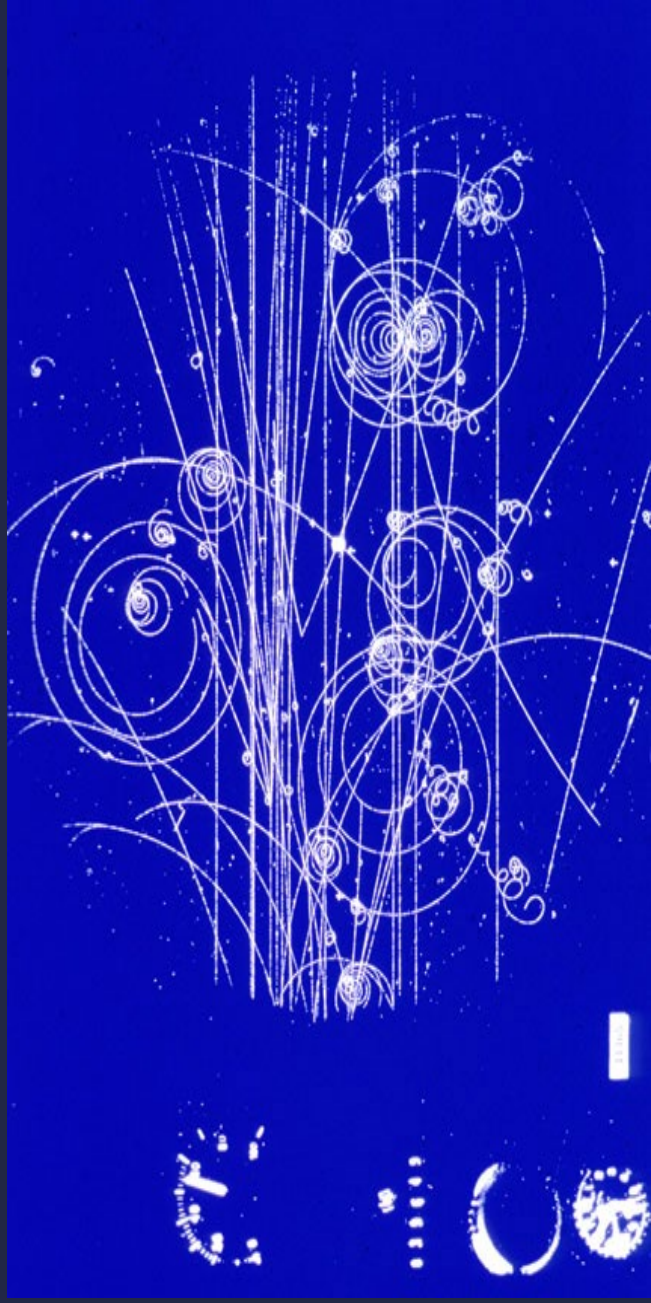
# Geant-X

- All attempts to develop the successor of Geant4 failed
  - Geant5
  - The GEANT
- All of them were independent of the Geant4 collaboration
- They tried to start from Geant4
  - Utilize existing code for physics process in Geant4
- They couldn't bring their achievements timely
- No more attempts are following
  - The reason why Geant4 could survive long



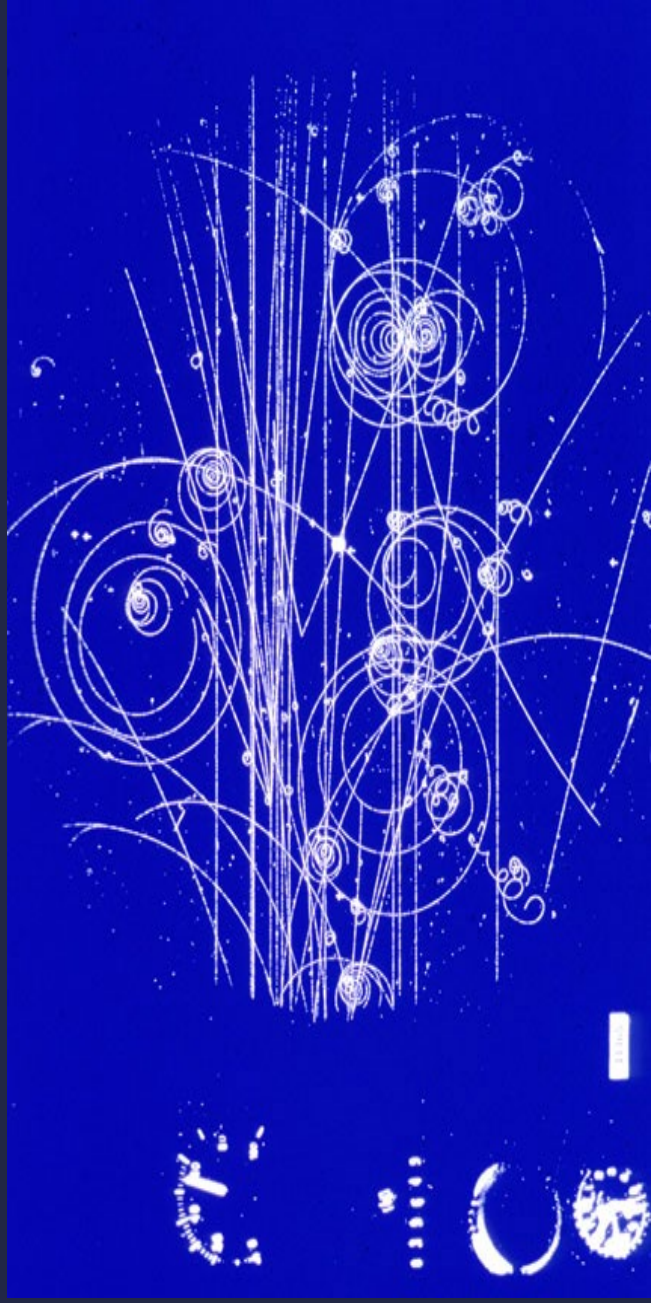
# Sociological aspects of software development in a large group

- Leadership
  - Efficient decision-making is the key to the success
  - Good leaders speed up the development
  - Core team (unofficial)
    - Driving force
- Organization
  - Democracy should be respected
  - How could we be very fair to latecomers?
- Collaboration management
  - Internal fights
  - FTEs

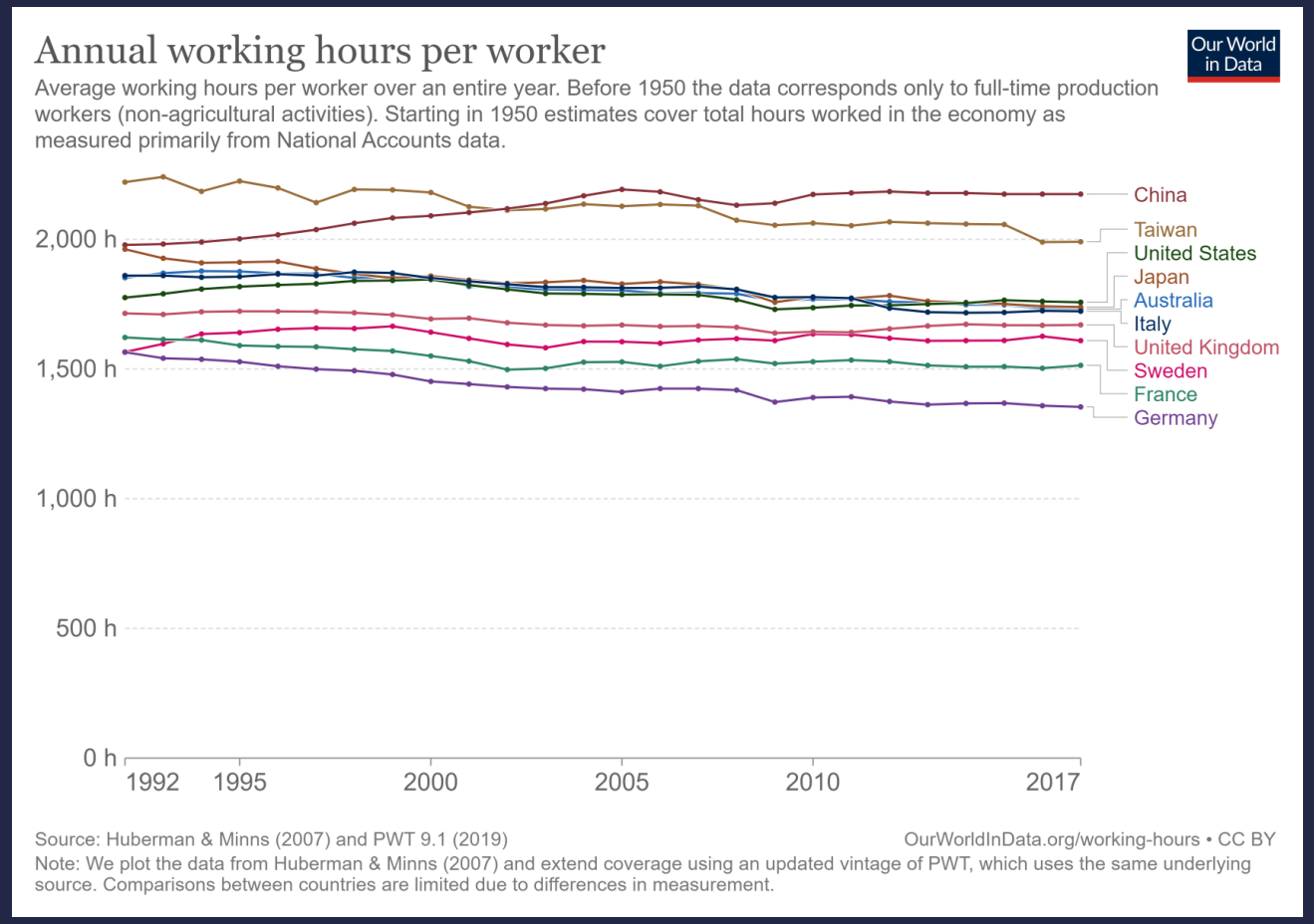


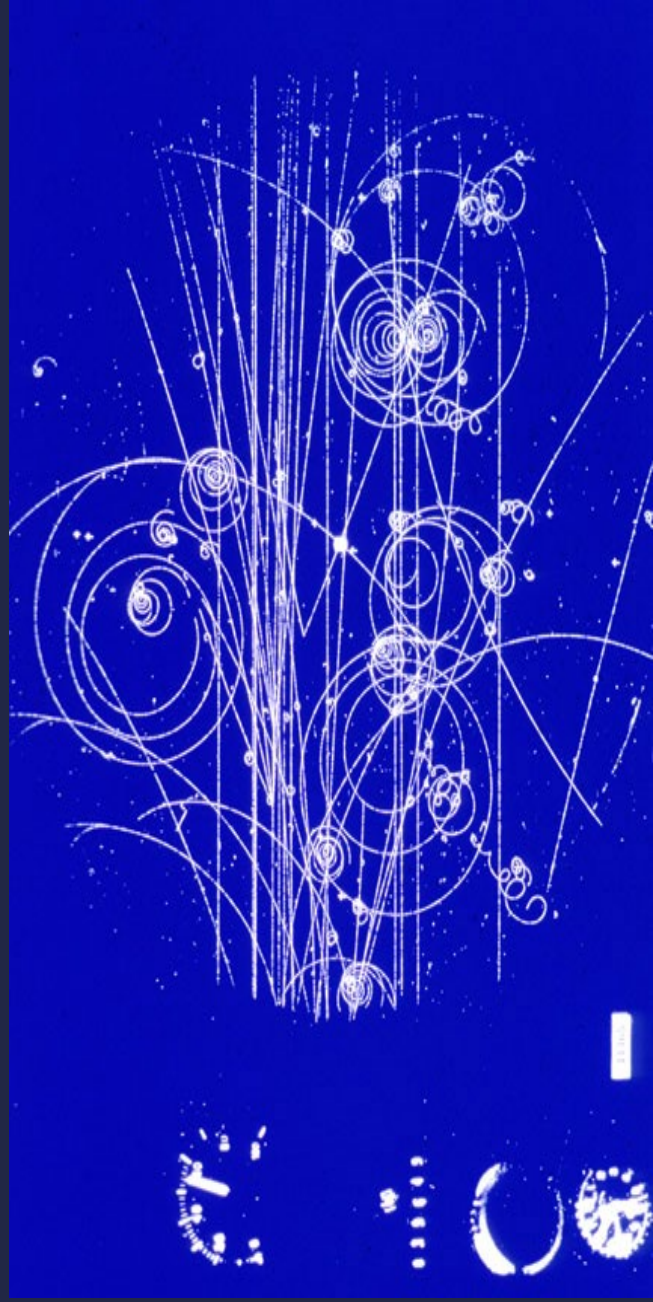
# Magic of FTE

- HEP collaborations evaluated the contribution of individual members and institutions by FTE in percent
  - FTE = Full-Time Equivalent
- How many hours are your “full-time”?
- Some declares 100%FTE for the project, but not realistic
- Concrete definition of FTE is necessary



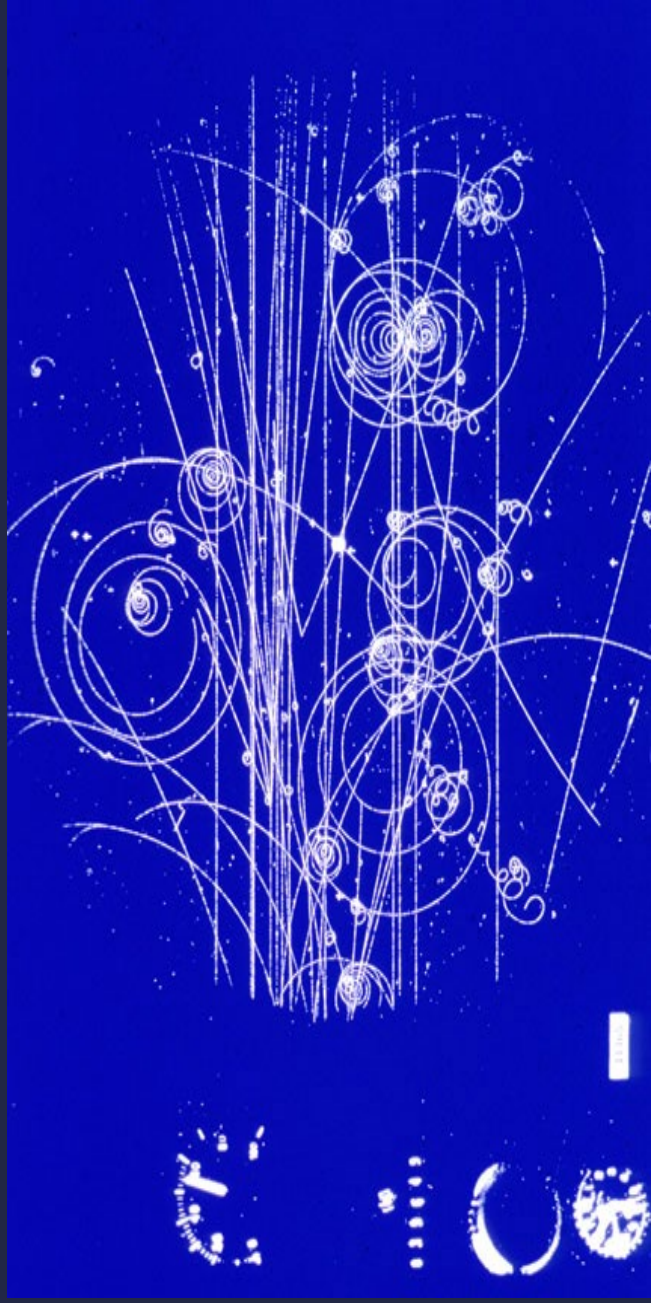
# Annual working hours



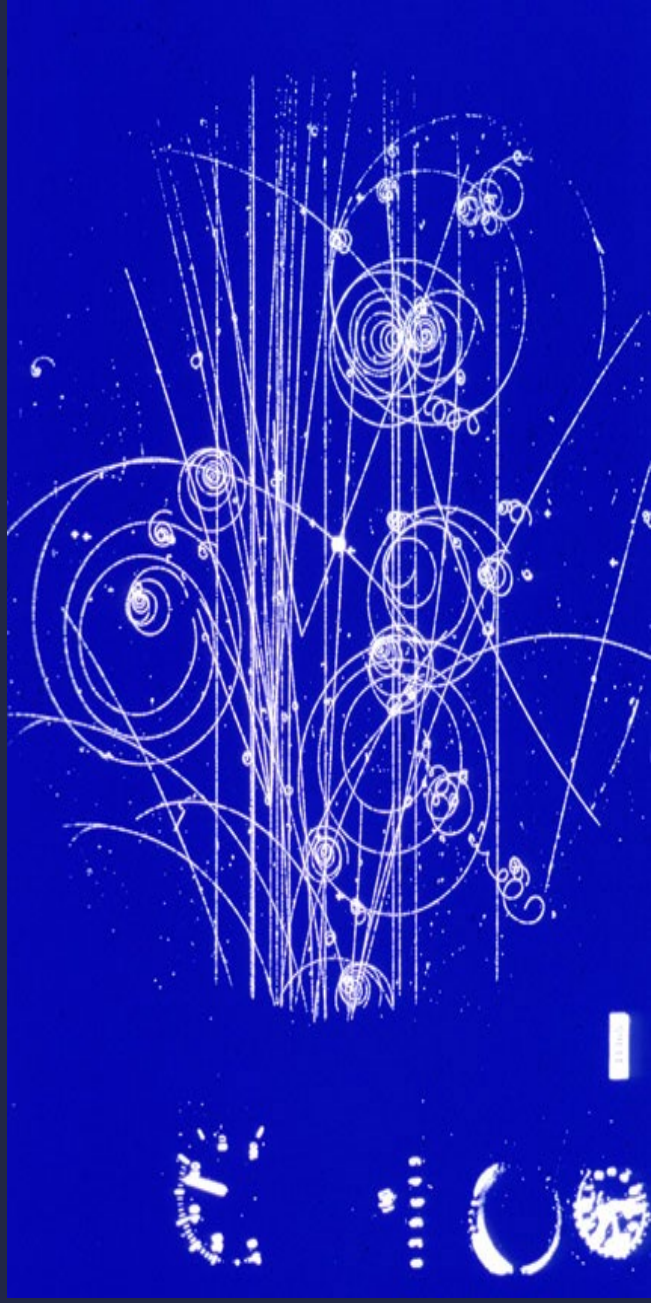
# Beautiful life

- The life of people in Europe looked like a dream for me
  - Comes late to their office, frequent tea/coffee gatherings, long lunchtime, and goes home earlier
  - Frequent leaves
  - Long Vacation
- I realized that my previous efforts to find a job in the USA were a terrible idea

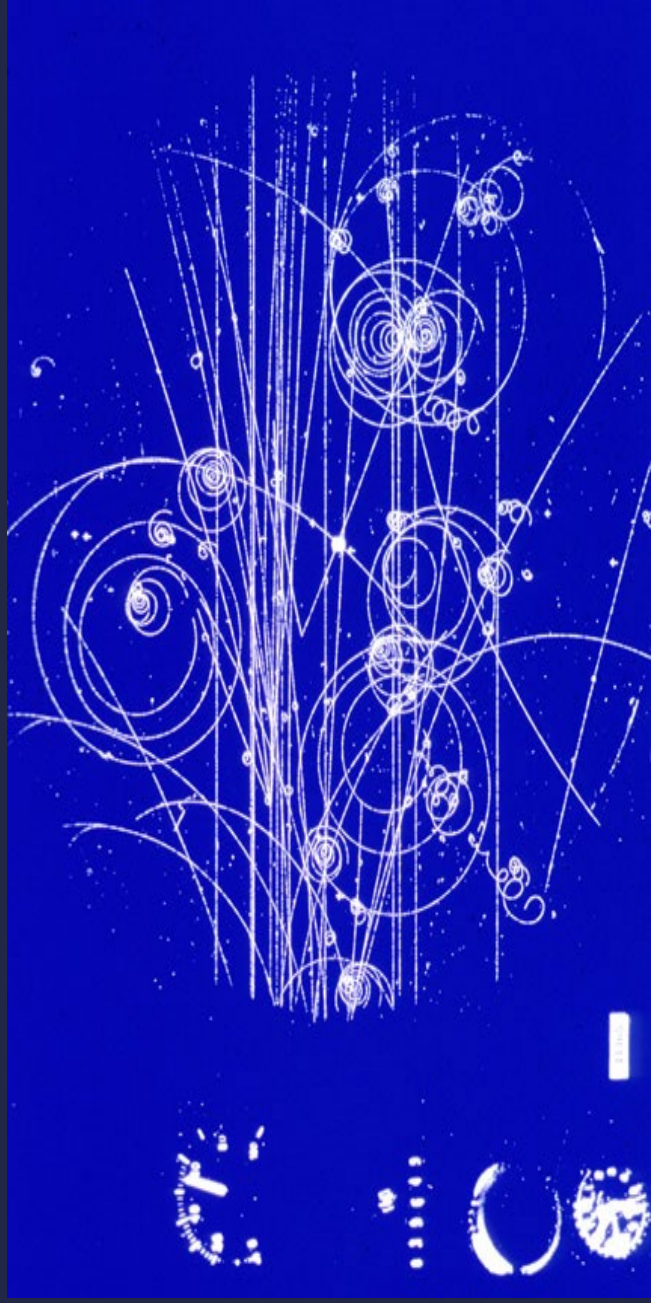


# Securing a job

- I found a type of developer who looked very active but ruined the project
  - No design
  - Commit the code with every minor change
  - No backward compatibility very often
  - Their codes are buggy very often, and nobody can maintain
- La raison d'être
  - Those people tried to demonstrate their ability to confuse others
  - Hard competition until obtaining a permanent position
- After those people leave, always huge mess happens

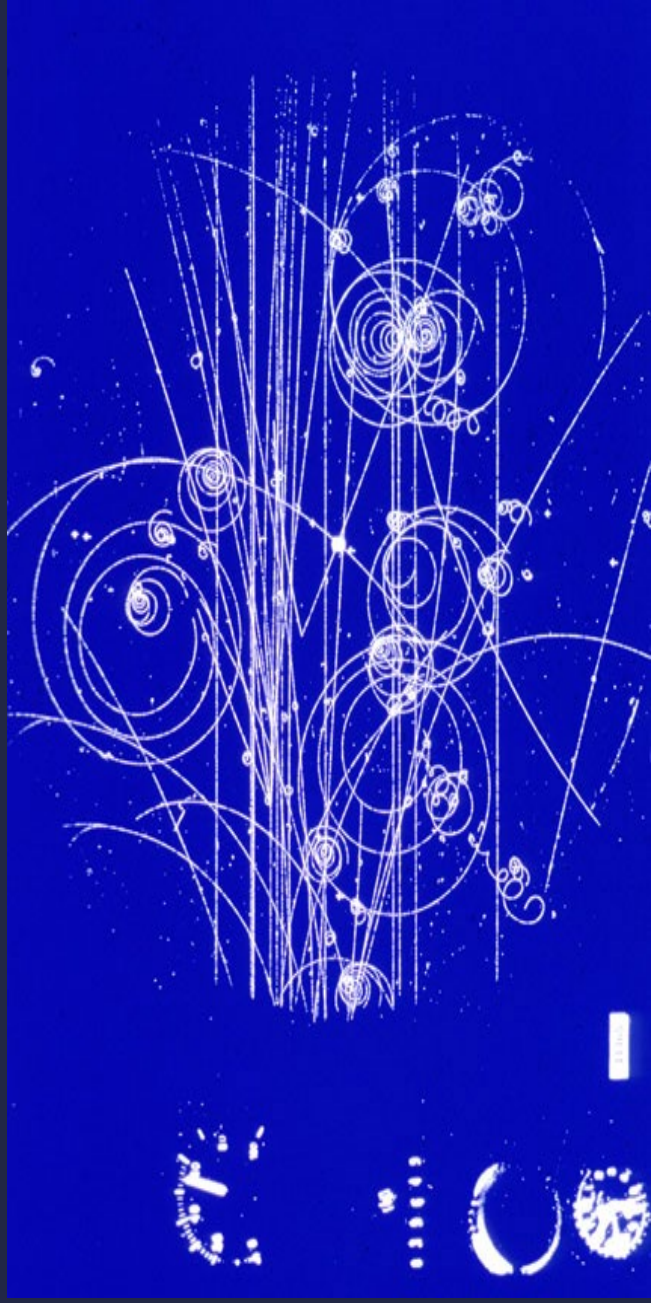


- Stable positions for young talented people
  - Too stiff competition among young people results in the waste of their workforce
    - Concentration on their research is necessary for good work
- Compared with the relaxed lifestyle in Europe, the research job situation looked to me more than the nightmare
  - Japanese situation is also abysmal



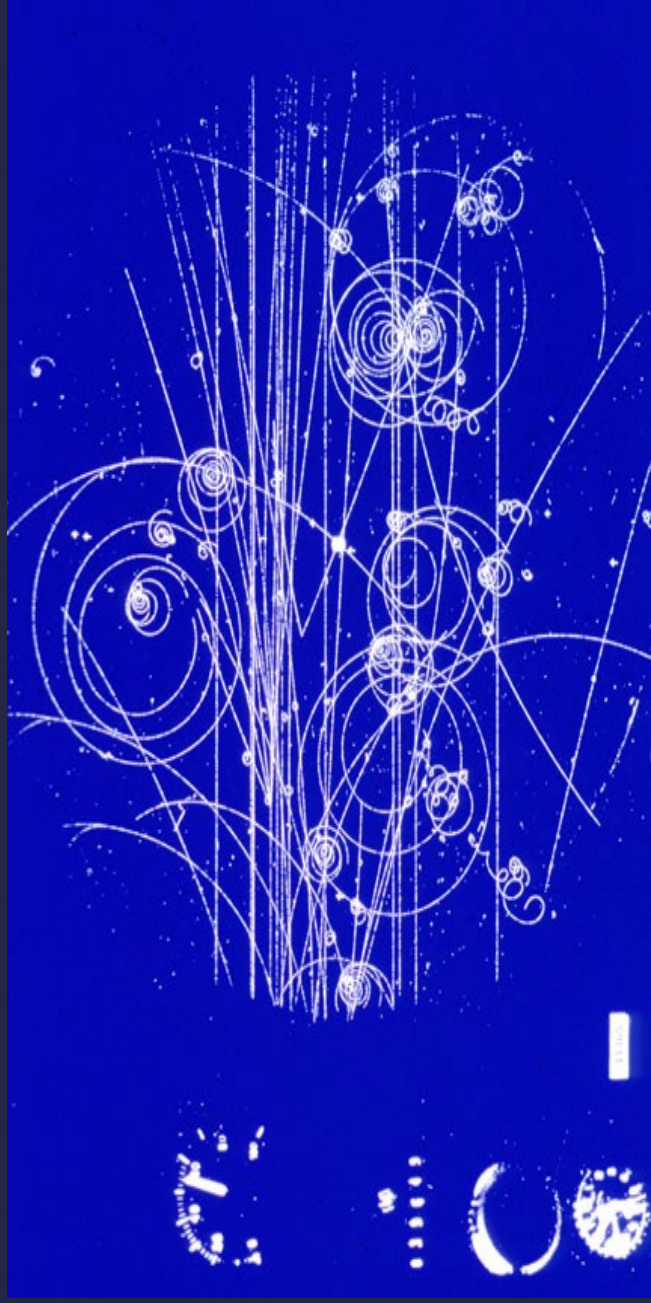
# RD44 days

- Mutual understanding and conquering cultural differences were not easy at the beginning
  - Atmosphere in the European collaboration is much different from the USA one for me
- Discussed the design for the first two years
  - Very tired
- Frustrated a lot because people outside attacked us a lot without evidence to say:
  - Programs written in C++ are very slow
  - C++ is not for physicists
  - GEANT3 is enough
  - Geant4 is much slower than GEANT3



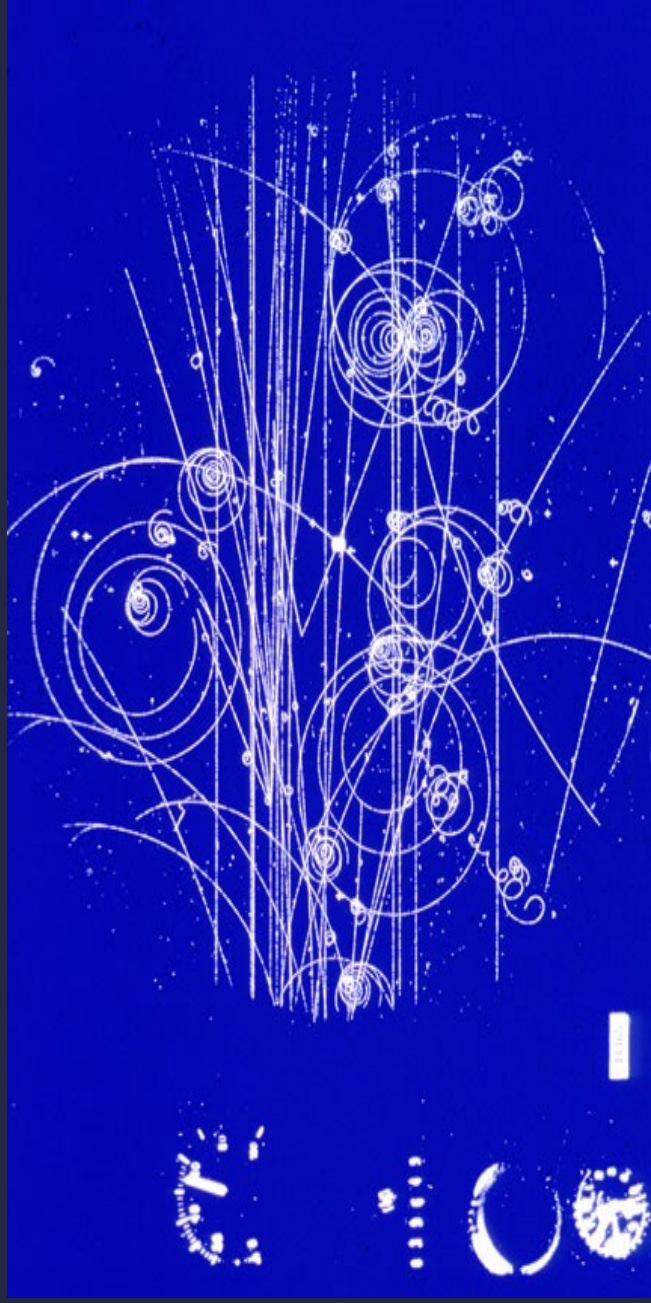
# Language (natural) in communication

- I heard lots of eccentric English accents (including Japanese) at CERN
  - Amazingly, people understood each other
  - The worst English language spoken to me was the Brooklyn accent until I went to CERN
- I stopped to practice pronouncing English words
  - Instead, I needed the training to hear others
  - Italian English emulation by Japanese works well for European people



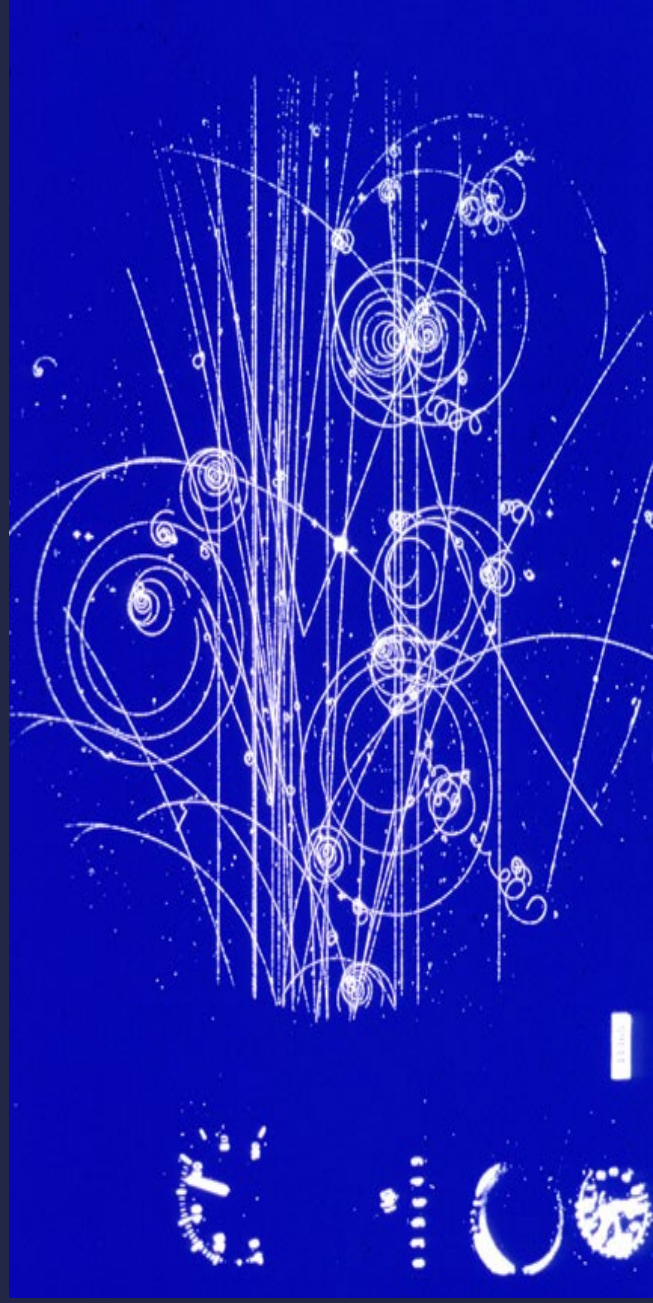
# “Saucisse, s'il vous plaît”

- Japanese developers visited CERN often during the design and prototyping stage of Geant4
- Soon after I arrived at CERN in the late evening on Sunday, which was my first visit, I was forced to speak in French at the cafeteria to make an order of food
- Learning French helped me to understand French people speaking in English
- Japanese pronounce “G4(じぃよん)” as “geant” in French☺



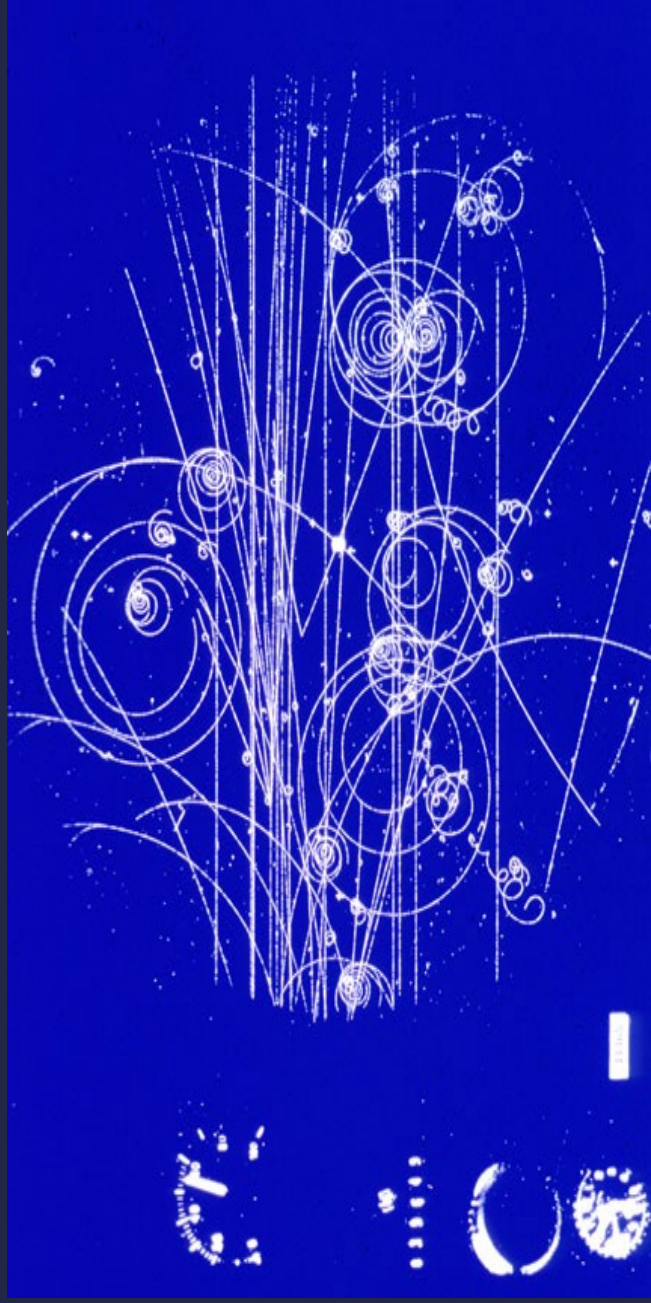
# Geant4 developers

1. Hired for Geant4 development and maintenance
   - IP belongs to the employer
   - They provide sustainable user support, not only development
     - CERN Geant4 team
     - SLAC, JLAB and FNAL Geant4 team in the USA
   - ESA contractors are somewhat special
2. Volunteers
   - Majority in the collaboration
   - Individual developer holds the IP
     - Reason why many individual names on Geant4 License
   - They often have positions at universities
     - They have the freedom to choose their research topic
   - Most of the Japanese developers are volunteers



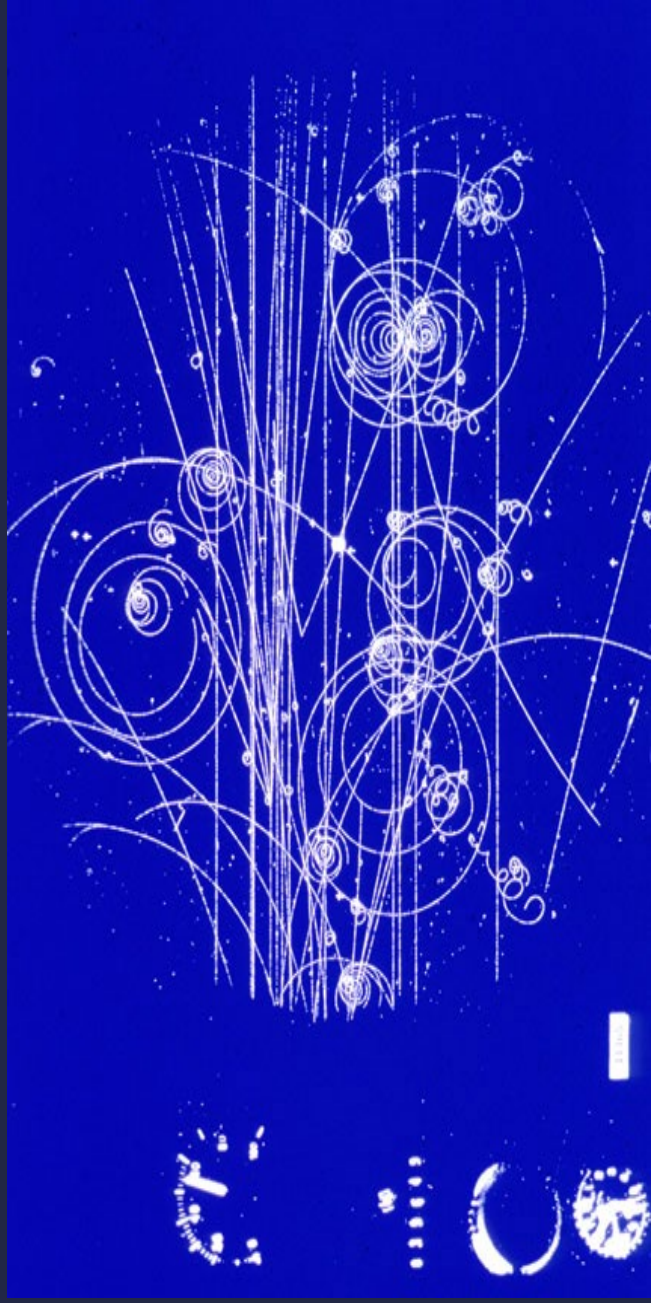
# Japanese Geant4 developers

- Mostly researchers in universities and KEK, and some postdocs and graduate students
- Works jointly with users for such as medical applications
  - Proton and carbon therapy simulation 2003-
- A new project, MPEXS
  - A CUDA simulator based on the Geant4 expertise



# MPEXS

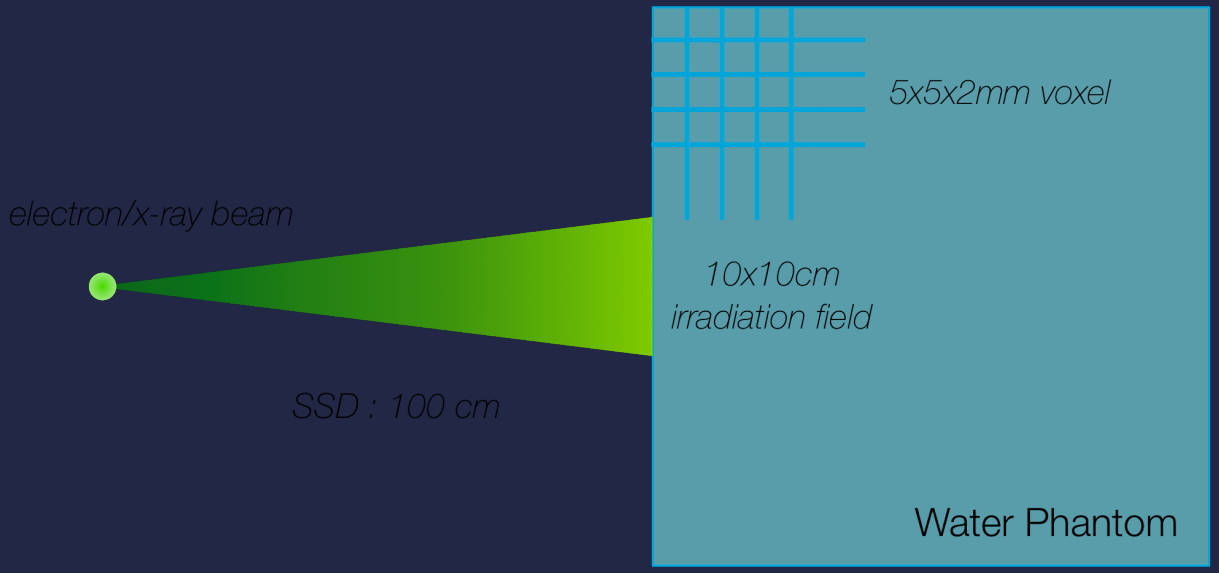
- A CUDA radiation transport simulator
  - Fully based on the Geant4 experiences
- Focused on non-HEP usage
  - Simple geometry but high statistics
  - Conventional physics only
    - No eccentric physics such as PeV muons
    - Code could be very stable
- Decided to go in the opposite direction from Geant4
  - Not an open-source project
  - Academic usage is free of charge, but the contract requested
  - Commercial license

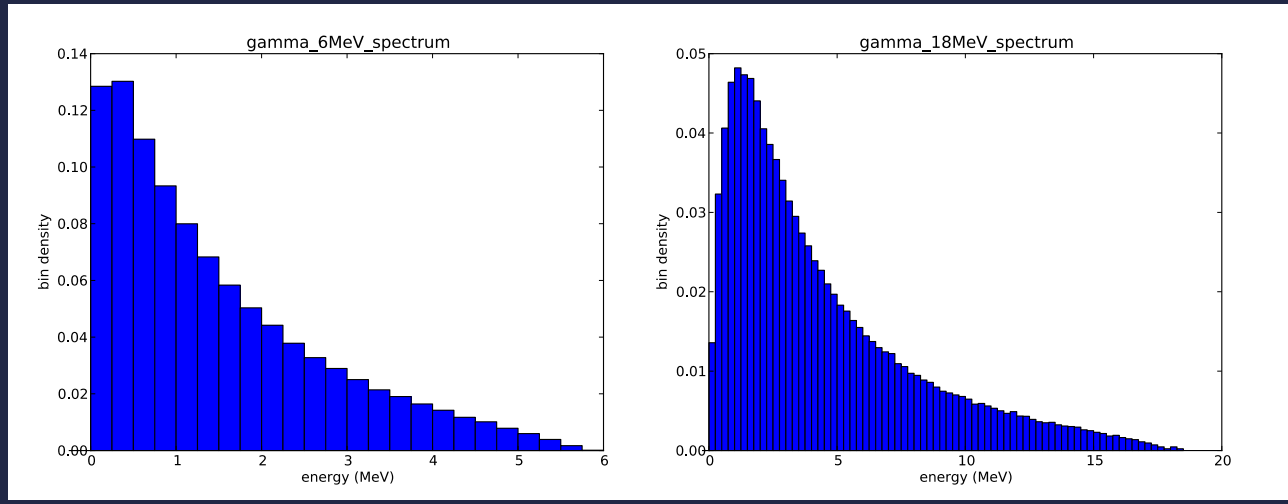


# MPEXS EM Physics Benchmark

Particle Sources

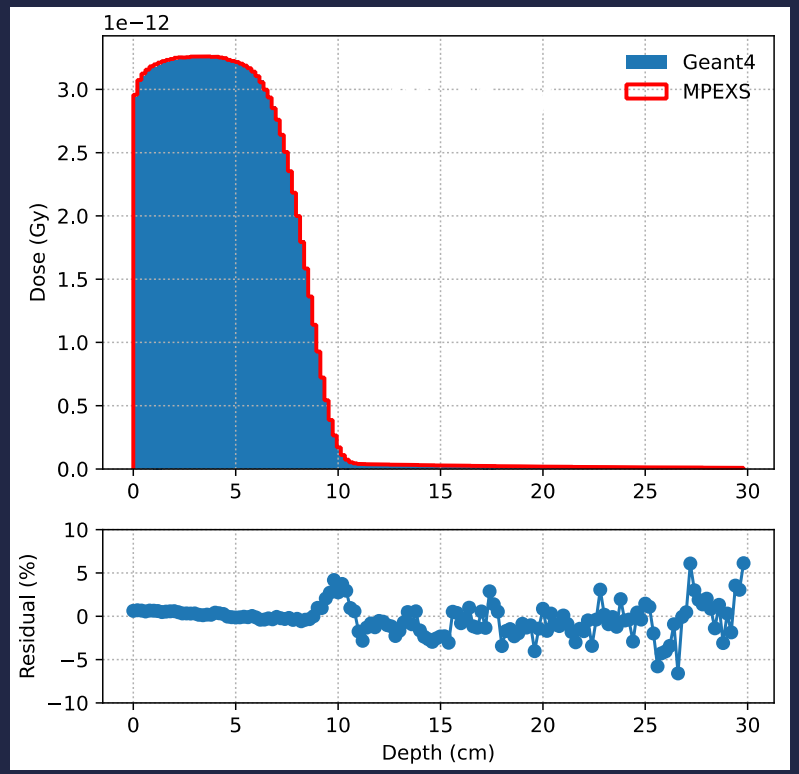
- SSD: 100 cm, Fields size: 10 x 10 $cm^2$
- Mono-energetic source
  - 20 MeV Electrons
- Spread energy source generated by medical Linac
  - 6 MV and 18 MV photons
- Irradiating water phantom with 500 M particles

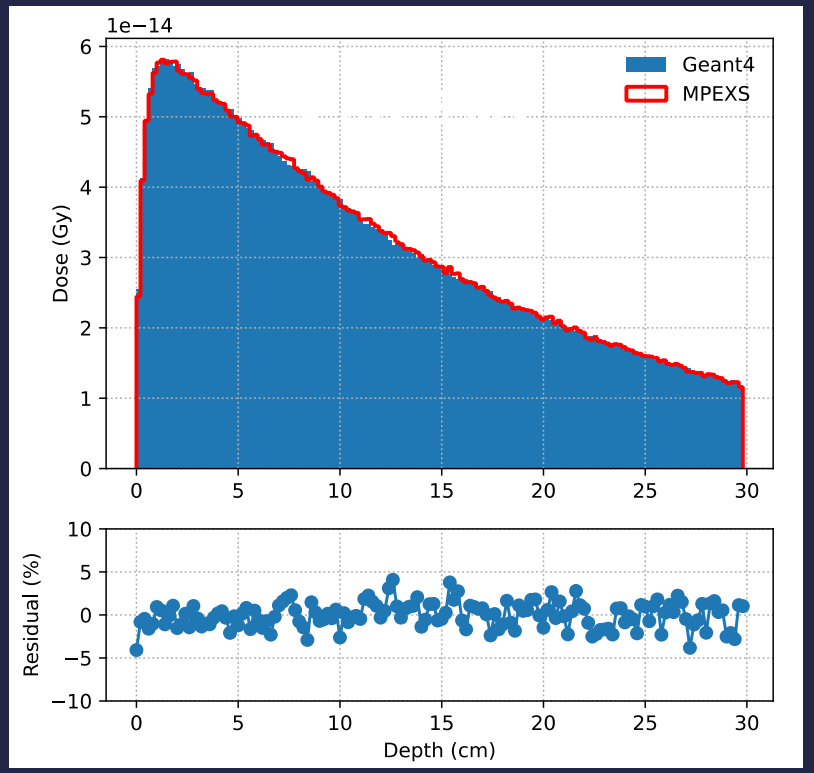


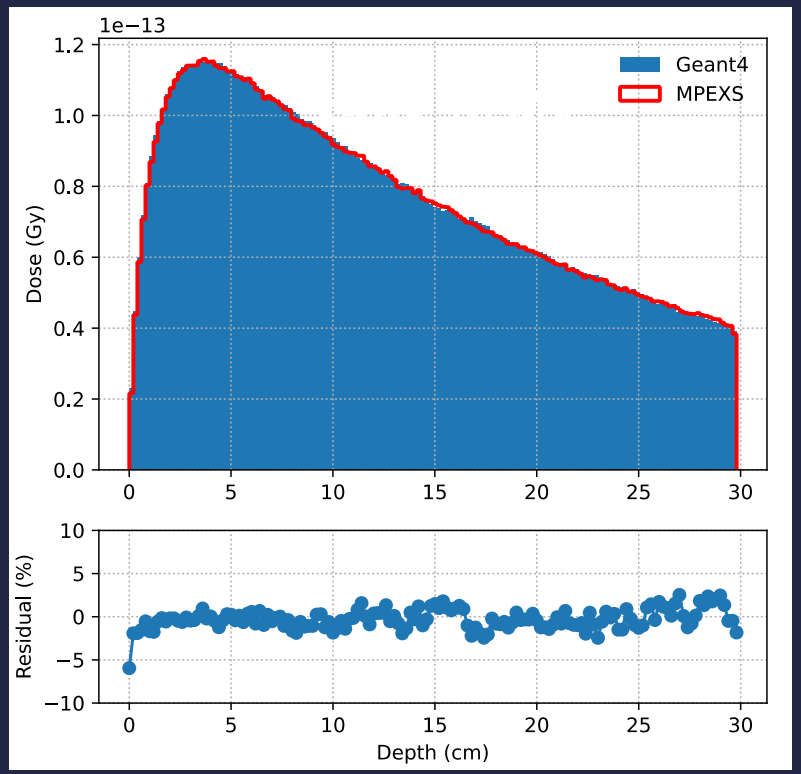



# Depth Dose Distribution for Water

## MPEXS reproduces Geant4



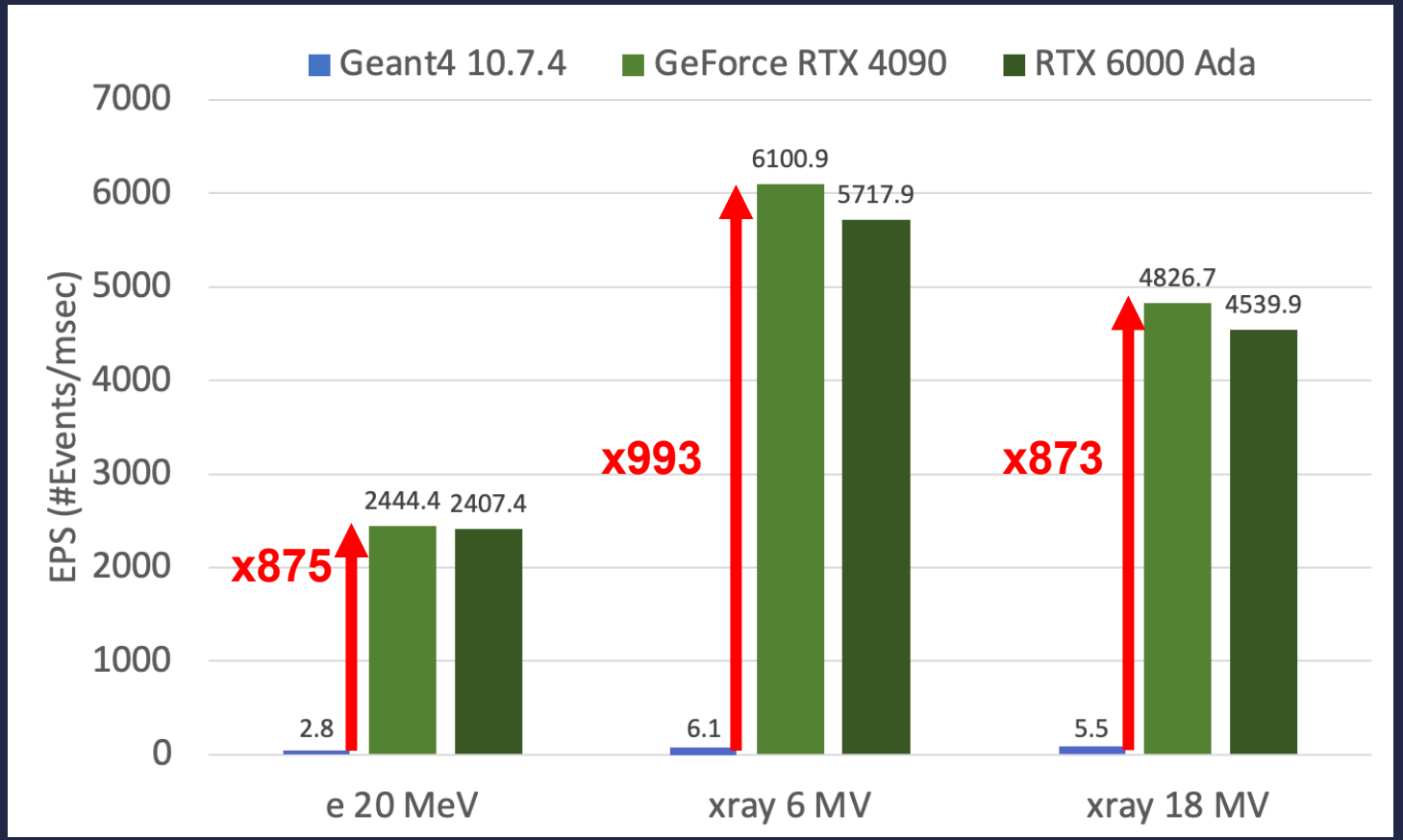

$$Residual(\%) = \frac{D_{MPEXS} - D_{Geant4}}{D_{Geant4}} \times 100$$

# MPEXS Performance

## Up to 1,000 times speedup against Geant4 with a single CPU core

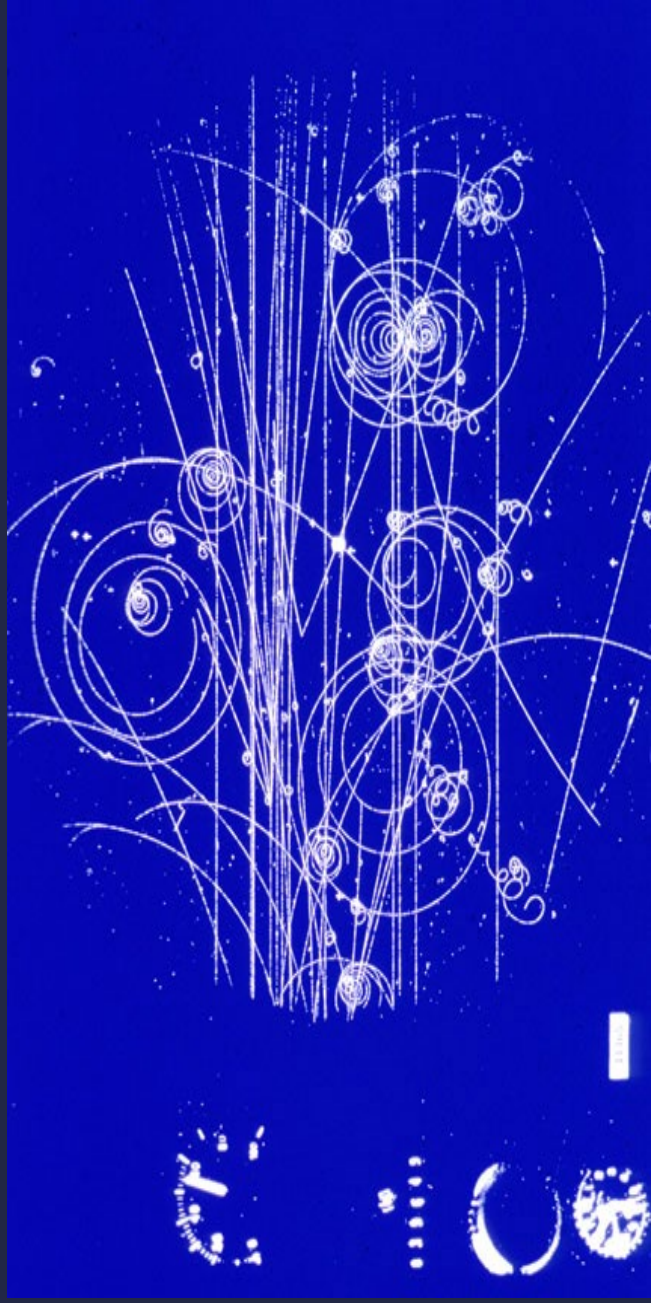
**Benchmark Environment**

- Alma Linux 9.1 (GCC 11.3.1 + CUDA 12.1)
- CPU
  - Intel Xeon Gold 6326 (Ice Lake)
    - 16 cores/32 threads, 3.5 GHz
- GPUs
  - GeForce RTX 4090
    - 16,384 cores / 2.52 GHz
    - Memory: 24 GB (GDDR6X)
      - Bandwidth: 1,008 GB/s
  - RTX 6000 Ada Generation
    - 18,176 cores / 2.50 GHz
    - Memory: 48 GB (GDDR6)
      - Bandwidth: 960 GB/s



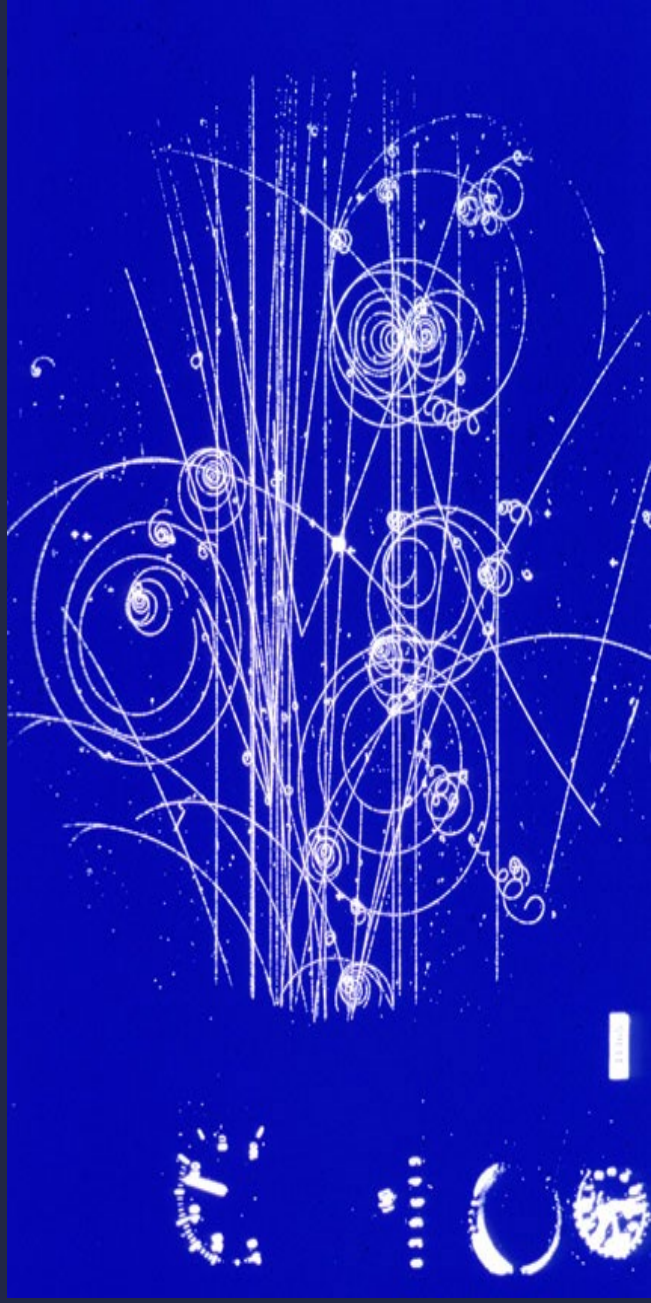
# MPEXS LLC

- A startup company founded on September 1st, 2022 for MPEXS commercial solutions
  - MPEXS2 series software
- A KEK venture
  - Exclusive contract with KEK to sell MPEXS commercial solutions
  - KEK doesn't have their own TLO
- Target medical users first



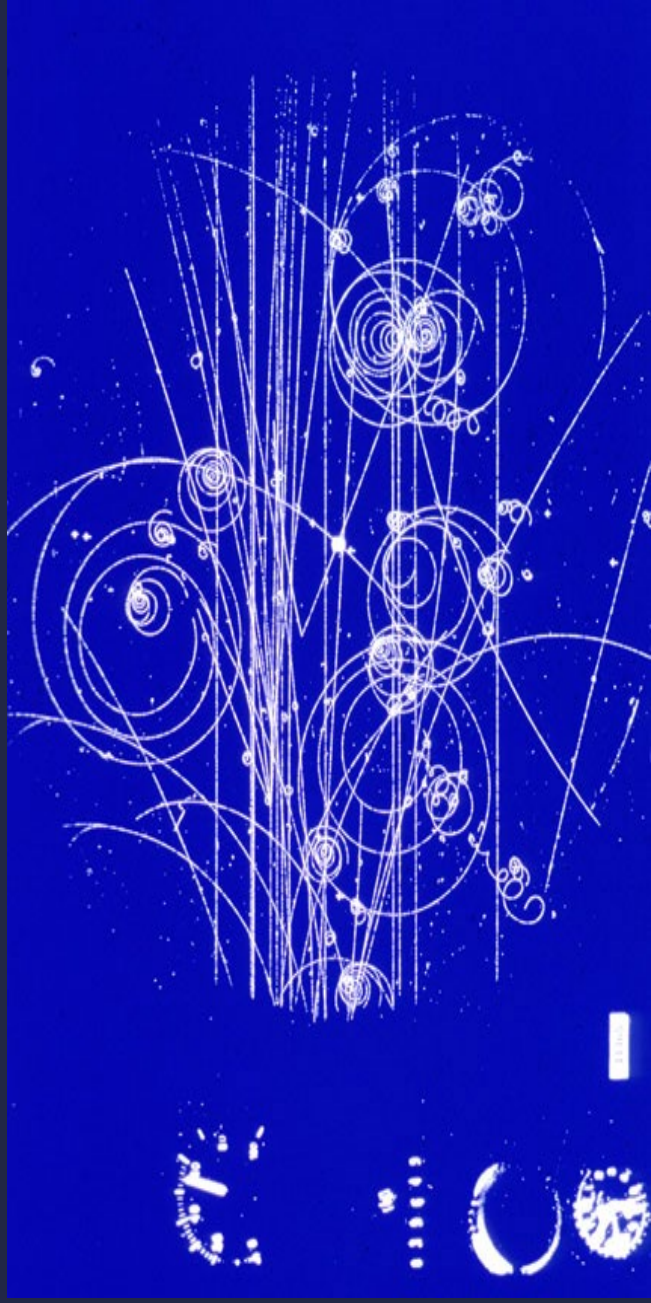
# Time flies

- Documentation
  - OOD diagrams and documents are no more maintained very well
- Geant4 is NOT GEANT4
  - FORTRAN only allows capital letters, but C++ also allows lowercase letters
  - Our new logo is all in capital letters
    - What a shame!



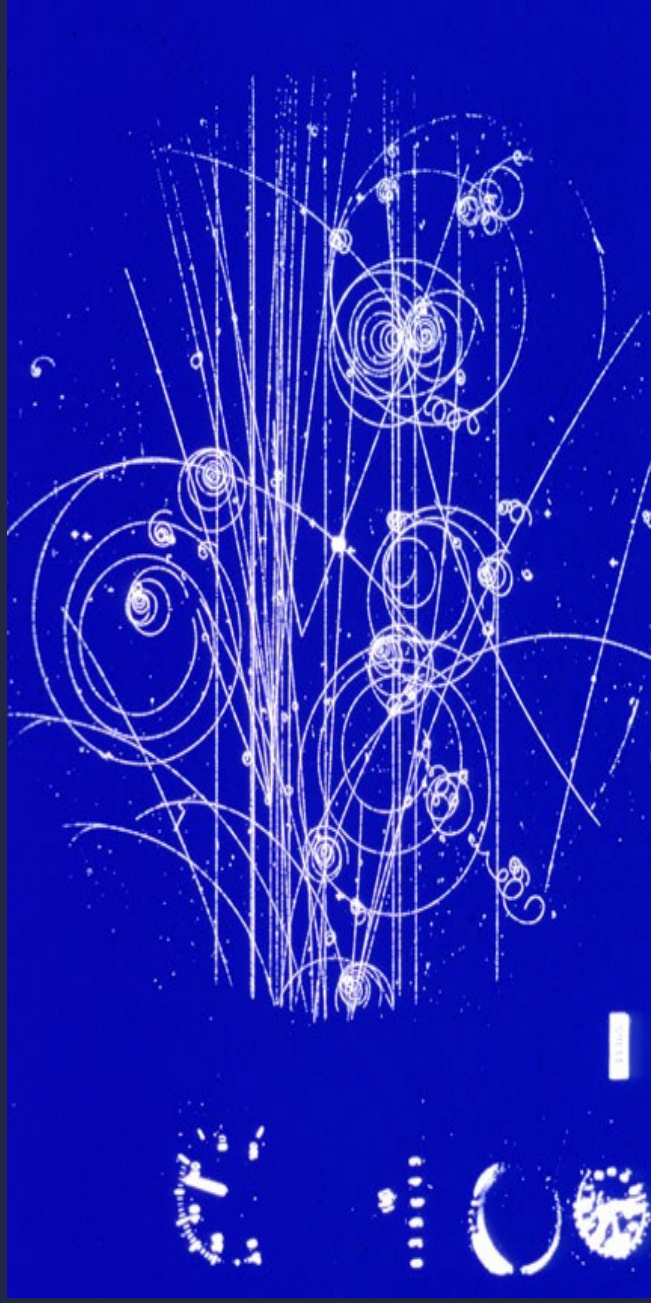
# COVID19

- We could work at home without any significant difficulties in the development
  - PC and Internet are essential
- Realized the importance of F2F meeting for mutual understandings
  - We need tight partnerships for successful development
  - Big decisions need a long discussion



# What are major difficulties in the long-life project?

- Funding
  - Funding agencies tend to fund a new project, but long-existing projects
    - They often say, "Maintenance of something is not research. We fund only research projects."
- Human resources
  - New young talents are always necessary for further development and enhancements
- Backward compatibility
  - Existing users don't want a significant change in Geant4 to be required a change on their side
  - Destructive development is necessary sometimes to follow computing environment trend



# Summary

- Geant4 endured for three decades with great success
  - We will celebrate the 30th anniversary of Genat4 the next year, 2024
  - Geant4 is the most successful radiation transport software
  - Further improvements and new functionality will come
- MPEXS, a new simulator on GPU, is available for medical physics and some other fields
  - More than 1000 CPU core equivalent computing power with 1 GPU unit (depends on the subject)
- Look forward to seeing the replacement of Geant4 very soon
  - The designed life of Geant4 will come soon