

The ATLAS SCT Production Database

A. Clark¹, Ph. Demierre², J.M. Dubois², D. Ferrère,¹ H. Nguyen², F. Serena²

University of Geneva

¹ DPNC : Département de Physique Nucléaire et Corpusculaire

² DINF : Division Informatique (DEVM)

July 26, 2002

Abstract:

The assembly of the Semiconductor Tracker (SCT) of the future ATLAS experiment at CERN must be achieved in 2004 before the final detector integration. A central database for SCT must trace the items of the module issued from the production with their relevant information like the test data of the quality control. A Database concept has been designed to allow the users from anywhere to access their data with simple tools. The accesses to the database items are given to any registered SCT user via internet. This note describes the concept of this database.



1. Introduction

The Semiconductor Tracker SCT, a sub-detector of the ATLAS experiment on the Large Hadron Collider (LHC) at CERN, will require important data storage during its construction. The SCT[1] is composed of 4088 instrumented modules based on silicon microstrip detectors, that will cover ~60 m², all around the proton-proton interaction point. The construction of SCT requires manufacturing in various institutes and industries all over the world. Also a relational database has been considered[2] and developed during the past years to centralize the information. An original development started in Manchester[3] and was based on local DB systems and the idea to mix and centralize the data to a central DB came later. This DB, hosted in the Geneva University, is flexible enough to allow accesses from various protocols like http, SQL net or JDBC. Due to this wide opening, secured accesses and privileges have been implemented to ensure a rather good security. The most common interface for this database is using a web browser like Netscape or Internet Explorer that allows consulting, insertion and update of the data depending of the roles that have been set by the DB administrator. To ease the insertion of mass of data, raw data and images a specific java tool has been implemented.

This note is intended to give a brief overview of the SCT production database and describes the specificities of the data structure, the definitions and the Graphical User Interfaces (GUI) supported by the Geneva University.

2. Concept

A database server is the key to solving the problems of information management. In general, a server must reliably manage a large amount of data in a multi-user environment so that many users can concurrently access the same data. All this must be accomplished while delivering high performance and availability. A database server must also prevent unauthorized access and provide efficient solutions for failure recovery.

The original choice of the developers was based on an evolutionary development DB model. It allows from an initial scheme implementation, and from user feedbacks to refining the DB structure through many version until an adequate system has been developed. This method is very useful to help the users to define their needs and essentially for defining the data structure.

The first data structure was quickly defined and frozen thanks to user feedbacks and was frozen soon after allowing a first application available in the production DB. Since this first version, the system is able to grow up and extend to the new needs of the users.

The whole application is based on a central database server that holds the data and the applications. The server allows secure accesses via wide available protocols like http or JDBC through the Internet. The concept has been designed to warranty a great reliability, robustness and maintainability, since everything is safely located and managed in a central place.

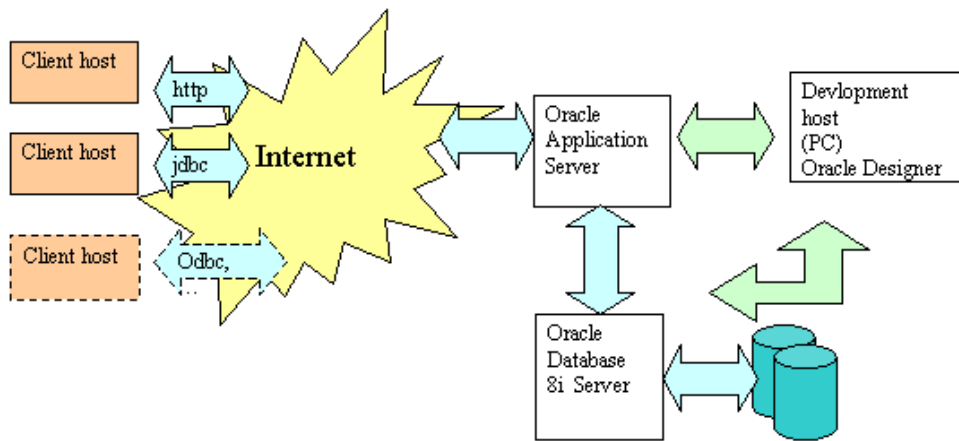


Figure 1: General SCT database scheme.

To reach the goal of project, two Oracle databases (8i) with application server (OAS4), running on a Solaris Sun Ultra Spark 450 were set up. The first database called development database is used for user validation purposes.

When the developers and few restricted users validate the new developments, they are implemented in the second database so called the production database and then it is made available to all users.

The tool mainly used for the data structure and the web applications is a Rapid Application Development tool (RAD) named Designer and provided by Oracle.

Additional Java client applications developed in Geneva and distributed from the SCT DB web page, were needed to allow the users to upload a large amount of test data in one step.

2.1. The development tools

The needs for the data management have been first discussed with the future users and, once it is agreed the data structure could be designed by defining the entities and the relationship between them. The Oracle RAD tool allows elaborating the entity relationship diagram that is the skeleton of the database. Once this design is considered satisfactory, this tool allows generating automatically the PL/SQL packages for the table creation and the various web views that will enable the data access. Despite the complexity of the data structure this RAD tool allows a quick, safe and bug-free development. The web Graphic User Interface (GUI) for which the code is generated automatically, is a static content with various forms, but the PL/SQL packages are responsible for accessing the data into the database and for wrapping the output results to a HTML page.

In addition to the web tools, several java applications have been implemented to ease the user mass data insertion. Java is a network and object oriented language. Its main advantage is that the application is entirely portable and so platform independent. The java SCT DB applications are materialized as JAR (Java ARchive) files that correspond to compressed classes and they are implemented using the Java Development Kit (JDK)[4].

2.2. The data architecture

The data structure can be designed as soon as all the user cases are identified. The design of the data structure is done on an Entity Relationship Diagram (see an extract on Figure 1).

The most widely accepted database model is the relational model and it has three major and general aspects:

- ❑ Structures : structures are well-defined objects (such as tables, views, indexes, and so on) that store or access the data of a database. The contained data can then be manipulated by operations.

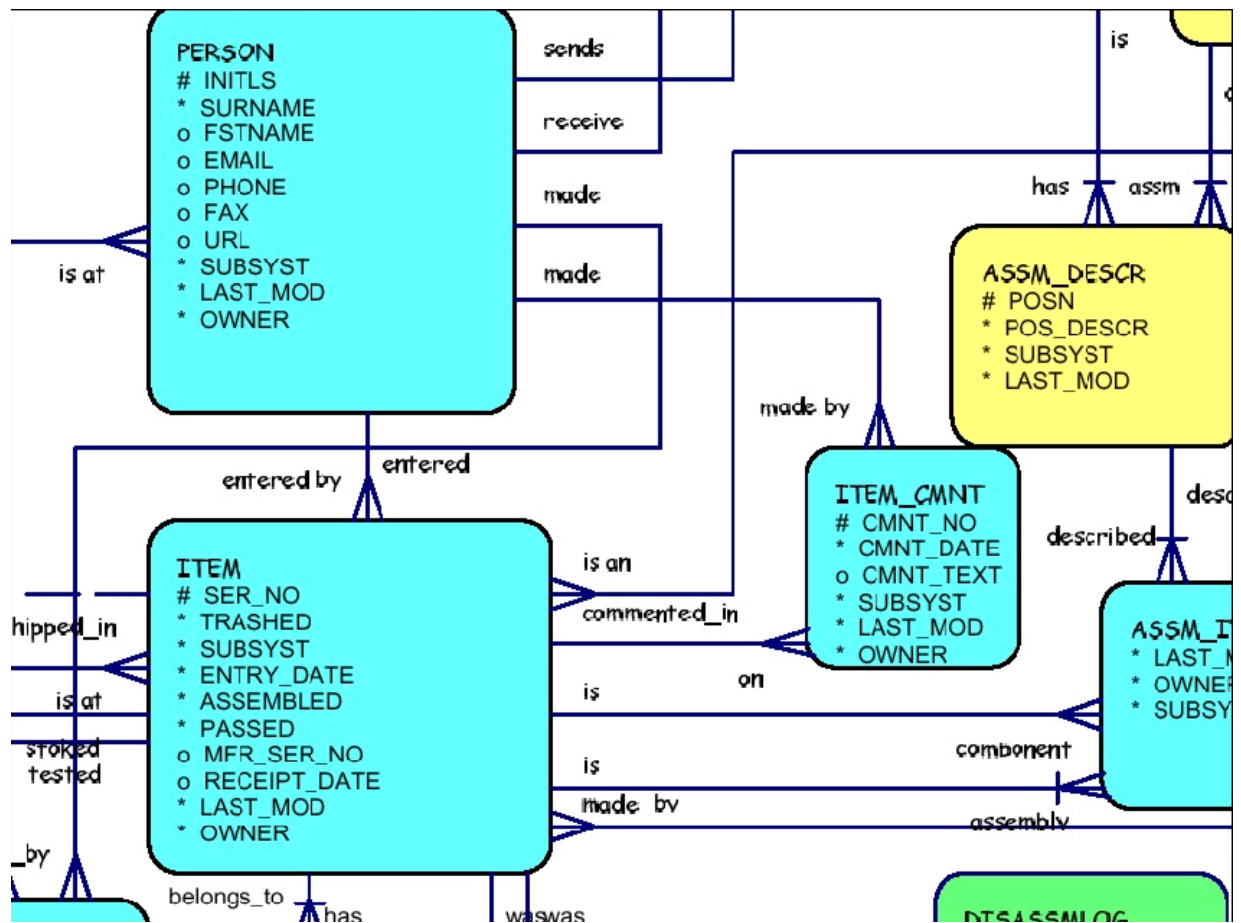


Figure 2: Expanded view of the entity relationship diagram for SCT.

- ❑ Operations : operations are clearly defined actions that allow the users to manipulate the data and the structures of a database. The operations on a database must adhere to a predefined set of integrity rules.
- ❑ Integrity rules : integrity rules are the laws that govern which operations are allowed on the data and the structures of a database. Integrity rules protect the data and the structures of a database.

In the case of the SCT database 3 classes of entity have been designed (see Figure 2):

- ❑ class 1 (in yellow): is the definition structure for the description of items, tests, assembly and is not accessible to the standard user but by one or few administrators.
- ❑ class 2 (In blue and red) : is the user data structure that should hold the physical item, tests data. Individual users own their data.
- ❑ class 3 (in green) : is a history structure for assembly when a deletion is made like for a disassembly. The data are stored automatically in a log table when a deletion is triggered. The data in the disassembly history log table are then only readable by any authorized users.

This database is an evolutionary model and, even if the core data structure is frozen for allowing partial production data entry, implementation of additional test tables is possible when requested by the users.

The design key of this DB is that the trace ability of items is as close as possible of the production reality. Every item is identified by a unique 14 digits serial number (see next section) and is located at a well-known registered location. Various operations on an existing item are allowed like tests, assemblies and shipments.

3. The SCT DB specificities

3.1 The serial number

A unique serial number will be used to identify various items that are issued from the CERN project. The basic rule of the item identifier is described in the reference[5].

2	0	2	2	I	I	I	x	x	x	x	x	x	x
---	---	---	---	---	---	---	---	---	---	---	---	---	---

The prefix of the 14 digits serial number shall code the origin of the item in the project (for instance 2022 is for ATLAS SCT). Then inside each subsystem, 3 digits (III) are reserved to identifying the location (or Institute) number from where an item has been registered. According to the SCT DB rule this must be lower than 098. The number 099 and above is reserved for any item registration made via the web interface. Then the last 7 digits (xxxxxxx) are free numbers that are used for the local item identification in each institute. Note that for the web registration the last 7 digits are not free and are sequentially generated.

Some items are defined as unique such as silicon detectors or chips but others like glue or SMD components are lot definitions. To get that information, the user can consult the item definition of the SCT DB. In general the batch items are defined with one single serial number except if shipment is made for a subset. In the last case, every subset will need its own serial number since it is defined as unique and indivisible.

One of the tools provided to users is the bar code generation allowing to display or/and print any item serial number that is registered. They are 2 special cases for which 2 to 3 bar codes are displayed:

- Barrel module assembly: it will generate the bar code of the module and the hybrid.
- Forward module assembly: it will generate the bar code of the module, the hybrid and the flex (see Figure 3)



Figure 3 : Bar code generation of a forward module

3.2 The definitions

The key for recording data into the DB, is to follow the definitions that have been put by the administrator. This definition rule is constraining to use standard naming with a minimum of possible mistakes. The table that contains definitions (can be seen as yellow in the entity relationship diagram) are:

- Item description
- Assembly description
- Test description
- Defect description
- Item category
- Test list
- Test limits
- Manufacturers
- Location

The various user data tables are using the key word of those definitions. For instance the location key word will be the “**Locn_Name**” and for instance the University of Geneva will have as key word : **Geneva**. In the table like for instance Items the location keyword will appear under “**Location Name**” field and when inserting a new record, a pop list will automatically appear with all the possible location names.

Due a long item list, a semantic is used in the item type key word (See **Figure 4**). The first 2 characters are defining the activity field and they are for instance:

- ‘bh’ : barrel hybrids
- ‘bm’ : barrel modules
- ‘ch’ : chips
- ‘fh’ : forward hybrids
- ‘fm’ : forward modules
- ‘oc’ : opto-components

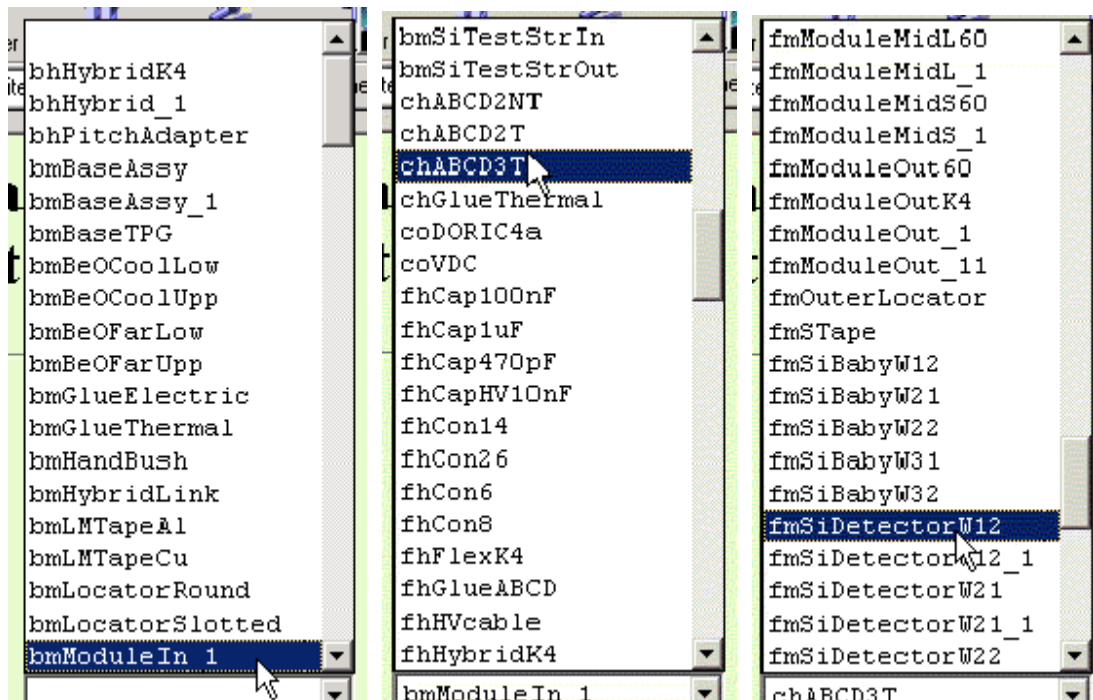


Figure 4: Some samples of the item definitions inside a pop list.

3.3 The events

An event or a set of events is always followed by an action on the DB like Insert, Update, Delete. A set of events exists to prevent the update or the deletion of data that does not belong to an institute. In addition to the intrinsic integrity given by the designed data structure, it allows keeping the integrity of the data by external rules.

In the SCT DB, specific events exist for shipment, assembly / disassembly and they are triggered as followed:

- Shipment validation: Set the shipped items with ITEMS.OWNER = sent, ITEMS.LOCATION = unknown and SHIP_ITEMS.OWNER = Destination Location
- Shipment reception: Set the shipped items with ITEMS.OWNER = Destination OWNER, ITEMS.LOCATION = Destination Location
- Assembly: Set ITEMS.ASSEMBLED = YES
- Disassembly: Set ITEMS.ASSEMBLED = NO, ITEMS.LOCATION = Location of the Assembly, ITEMS.OWNER = OWNER of the assembly, fill ASSEMBLY_HISTORY

Some events are triggered when the following actions are performed:

Shipment validation → Update of “confirmation date” in SHIPS table

Shipment reception → Update of “receive” in SHIP_ITEMS table

Assembly → Insert of component item in ASSEMBLY table

Disassembly → Delete of component item in ASSEMBLY table

4. The user interfaces

In addition to the physical database the access tools must allow easy and efficient access tools. Also essential tools and functionalities have been implemented to ensure that the data can be entered in an easy way by any user and from any platform. Some web basic reports do exist as well but detail analysis using information in the database is left to the various SCT activity groups to implement. Therefore a specific read only access has been created allowing any authorized developers to implement their own analysis tool.

4.1. The web interface

The web access, via the navigation panel of the welcome SCT production DB page[6] allows the consultation, insertion, update of data depending of the user privileges. A Read-Write access is opened for all the SCT users that have its own Institute username and password. The administration access is restricted to few people that are responsible to define the tests, items and assemblies related to their own activities.

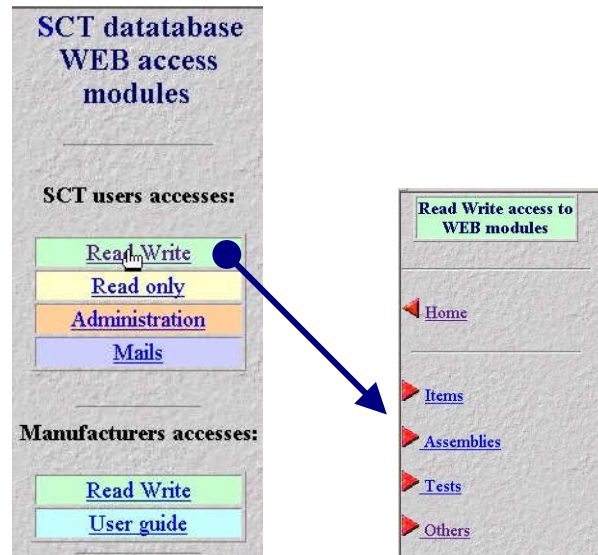


Figure 5: Extract of the navigation panel for accessing the table data.

Items	Assemblies	Tests	Others
Item description, Event description, Generic items, Chips, Detectors, Shipment, Item Events, Basic reports.	Assembly description, tree reports, Assembly, Chip assembly, tree reports, history.	All test description, (Test, defect, list) All detector test access module, XYZ module survey, Low mass power tapes, ... Basic reports.	Location, Manufacturer, Person.

Table 1: Content of the SCT categories in the navigation panel (in Red: are read only access; In Green: are insert, consult, update and delete access).

4 groups of data access are proposed: Items, Assembly, Tests and Others that are described in the Table 1. The direct access to the table data is then possible and follows the tree data-structure. In addition, some accesses to basic reports are in addition available and allow displaying a group of data resulting from particular queries.

4.2. The data upload using java applications

Insertion of measurement data sheet, raw data files and images directly into the SCT production DB is possible using java applications. The java tool is complementary to the web GUI that allows doing only manual insertion of attributes with no possibility to insert raw data or images. The applications packaged into JAR files are widely distributed on the web[6] and need to be downloaded. It works on any platform for which a Java Runtime Environment (JRE) has been locally installed. All the distributed jar files and the detail commands are described in Table 2.

In addition to the standard data file when filling the DB tables, raw data and image files (in JPG or GIF format) can be uploaded as well. Every time the users wish to upload a set of files, they will have to run the application by executing the command: ***“java UploadTestData”***. A GUI is then displayed to the users (see Figure 6) for inserting either a specific file or all the files from a directory (*.txt). The application will then generate locally a log file and will insert the data with the attached files into the relevant places of the DB.

The specific ascii template files for the upload are available on the documentation of the SCT production web page[6].

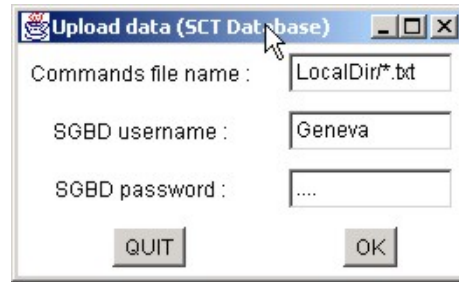


Figure 6 : GUI of the java application for uploading tests

<i>Description</i>	<i>Command</i>	<i>JAR distribution</i>
Upload test data	Java UploadTestData File SCTusername SCTpassword	SCTTEST_v_xx_P.jar
Upload item data (choice of SerNo)	java UploadItemData File SCTusername SCTpassword	SCTTEST_v_xx_P.jar
Upload shipment data (need web for trigger)	java UploadShipData File SCTusername SCTpassword	SCTTEST_v_xx_P.jar
Upload item image	java Upload_ItemImage SCTusername SCTpassword SerNo file.gif -T my image title -U http://www.cern.ch	SCTTEST_v_xx_P.jar
Upload test image	java Upload_TestImage SCTusername SCTpassword TestNo file.gif -T my image title -U http://www.cern.ch	SCTTEST_v_xx_P.jar
Download raw data	Java DownloadRawData TestNo InFilename SCTusername SCTpassword OutputFilename	SCTTEST_v_xx_P.jar
Download last raw data	java GetLastRawData SerNo testname outfilename userAccess	SCTspecialDev.jar
Extract chip SerNo and its wafer position	java GetChipPosSerNo WaferMfrSerNo ChipType OutputFile Username	SCTspecialDev.jar
Upload chips and its wafer	java UploadABCDChips File SCTusername SCTpassword	SCTABCDCHIPS_v_xx_P.jar
Upload assembly data	javaUploadAssmData File SCTusername SCTpassword	SCTAssm.jar

Table 2: All the possible java applications that are distributed and maintained.

4.3. Web basic reports

The web basic reports allow the user to access widely the data of various tables and summarize the information on a simple layout page or table. The difference with the specific web access module is that it is not restricted to a specific table view. Nevertheless a set of query field should be entered or leave blank by the user. The list of available basic report is summarized in Table 3.

Description	Found under
Item basic report	Item
Detector delivery status	Item
Shipped Item	Item
Assembly basic report	Assembly
Assembly description tree report	Assembly
Assembly tree report	Assembly
All Tests Report	Test
Institute IV report	Test
Defect report	Test

Table 3: List of basic reports that are available.

4.4. Other specificity

- **Generic assembly**

According to the assembly definition (see for instance Figure 7 for the description of the outer forward module) the serial number of an assembly such as a module and its components must be entered either via the web interface or using the specific java application: ***UploadAssmData***. In the case of the module, the position #1 and #2 correspond to the top and the bottom side respectively. Before an assembly record is inserted, the user must be sure that both the assembly and the component items are already registered into the item table with their own serial number. Then each assembly record will consist of entering the assembly serial number, the component serial number, the numerical position and the assembly date. Note that if a unique item such as detector is already assembled or if the user is not the owner of that item, the DB will refuse a new insertion

fmModuleOutK5	
Component type	Pos.
fmHybridK5	1
fmFaninOutL	1
fmFaninOutL	2
fmFaninOutR	1
fmFaninOutR	2
fmGlueElectric	1
fmGlueMechanic	1
fmLocationPad	1
fmOuterLocator	1
fmSiDetectorW31	1
fmSiDetectorW31	2
fmSiDetectorW32	1
fmSiDetectorW32	2
fmSpineOut	1

Figure 7 : Assembly description of the outer K5 module with the position of the items.

When using the java application by executing the command “***java UploadAssmData CommandFileName SCTusername SCTpassword***”, the format file must be as followed:

```
%Assembly
ASSEMBLY ITEM: 202209900000826
# CompSerNo      Position      Date (DD/MM/YYYY)
202209900000002  1          28/09/2001
202209900000001  1          28/09/2001
202209900000010  2          28/09/2001
202209900000011  3          28/09/2001
```

- **Chip assembly**

It has been agreed inside the SCT collaboration that the chip tracking must be reliable and every chip must be stored into the DB. At any step of its life, any user should know where a particular chip is located, either assembled or stored. As described above the SCT DB relies only on the item primary key, which is the 14 digits serial number. Since there is no possibility to identify and label individually the chips, there is in addition to the serial number, the XY position information on the wafer. Therefore when the chips are cut, the one that passed the quality control are stored in gel-packs at a certain and know position. The information is inserted into the assembly table of the DB thanks to the java application and an automating program that has selected the good chips. Every gel-pack is identified with its own serial number even if it is recycled.

In the spirit of easing the chips to hybrid assembly record a special web application was implemented. This DB operation would consist, if it was manual entry, of deleting individually the chips from the gel packs and reassembling them onto the hybrid at a certain position. This simple web tool requires that the user fill the gel-pack serial number, the hybrid serial number, the position of the chips in the gel-pack and, the position of the chip on the hybrid. Less confusion is possible using this tool and it avoids losing information of the previous position of a chip from an assembly.

Assembly Serial Number:	20220990004161	(Mandatory)
Gelpack Serial Number:	20220990304145	(Mandatory)
Assembly Position #	Gelpack Position # (Optional)	Assembly Date (Format: DD Mon YYYY)
1	10	Today
2	11	Today
3		Today
4		Today
5		Today
6		Today
7		Today
8		Today
9		Today
10		Today
11		Today
12		Today

SCT Database - Chip Disassembly/Assembly Action

The following records are under deletion (This action is equivalent to remove chips to gelpack):

Gelpack Ser No	Chip Ser No	Position #	Assembly status
20220990304145	20220990304151	10	Deleted
20220990304145	20220990304152	11	Deleted

The chip to hybrid assembly records are:

Gelpack pos #	Assembly Ser No	Chip Ser No	Hybrid position #	Assembly date	Status
10	20220990004161	20220990304151	1	18-FEB-02	Assembled
11	20220990004161	20220990304152	2	18-FEB-02	Assembled

SUCCESS !!!

Action made by W3SCTDEV
18-02-2002 12:51:02

Figure 8: Web GUI for chips to hybrid/module assembly.

- **Display test status function**

Basically 2 status fields have been implemented into the DB: ITEM.Passed and TEST.Pass. Only test passed is used and recorded by the acquisition system depending if a particular test of an item passed the criteria. On some of the items like detectors or modules, not only a single test is performed but a series of tests that can be performed at different time and by various institutes. Therefore a function, Gettestpassed(SerNo), has been implemented that allows checking when it is called the test status of an item. All the various optional and mandatory tests that have been

performed are scanned and the function checks the TEST.Pass status for each and makes a product of the Boolean field.

This function will return a string with a red or green color giving the overall status of an item. The possible answers are listed below:

- “All required tests performed and OK!”
- “All required tests not yet performed, and OK!”
- “All required tests performed and failed!”
- “All required tests performed and OK! Optional Test performed Failed!”
- “No test list defined!”

- **Raw data display**

The raw data are uploaded into the TEST_RAWDATA table and the attribute RAWDATA is a long raw type that is not visible in a directly on the web. A java applet has been implemented allowing the user in front of a web GUI to display the raw data (See Figure 9). As an alternative of the download, the raw data can be selected with the mouse and copied to another place on the local computer.

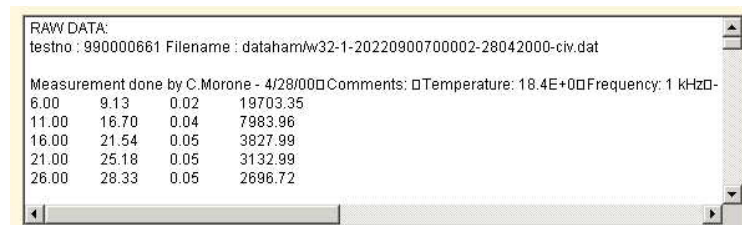


Figure 9: Web raw data display lunched by a java applet

4.5. Detector manufacturer access tools

In the case of the detector production, specific rules[8] were agreed to allow easing detector and test registration already at the manufactory. One of the major rules was to give access of the database to the manufacturers by keeping the confidentiality of the data. The second main rule was to allow a specific semantic in the 14 digits serial number easing the logistic and the human bar code reading.

Each detector manufacturer got its own DB to insert and visualize their own data with a number of restricted web and java tools. The access modules are limited to the following actions:

- Registration of detector into item table using a java application,
- Consultation of items using a web report that are owned and still at the manufactory location,
- Shipment registration and consulting using a specific web module,
- Insertion the manufacturer test data sheet using a a java application,
- Consultation of the entered test data sheet that is owned by the manufacturer.

5. Integration into data acquisition system

As described above, if a java application is used it will allow the users to upload test data from local files to the DB. A considerable advantage of such application is the possibility of automating a test procedure (see Figure 10) using Labview or any acquisition system like SCT DAQ[7].

The acquisition software based on any languages like C++ or Labview can then control and readout the Device Under Test (DUT). It will extract the relevant information in a predefine format file for the upload. The typical tests that need to be uploaded during the assembly are the individual

detector IV scan, the module IV scan, the XYZ module survey and the SCT readout tests.

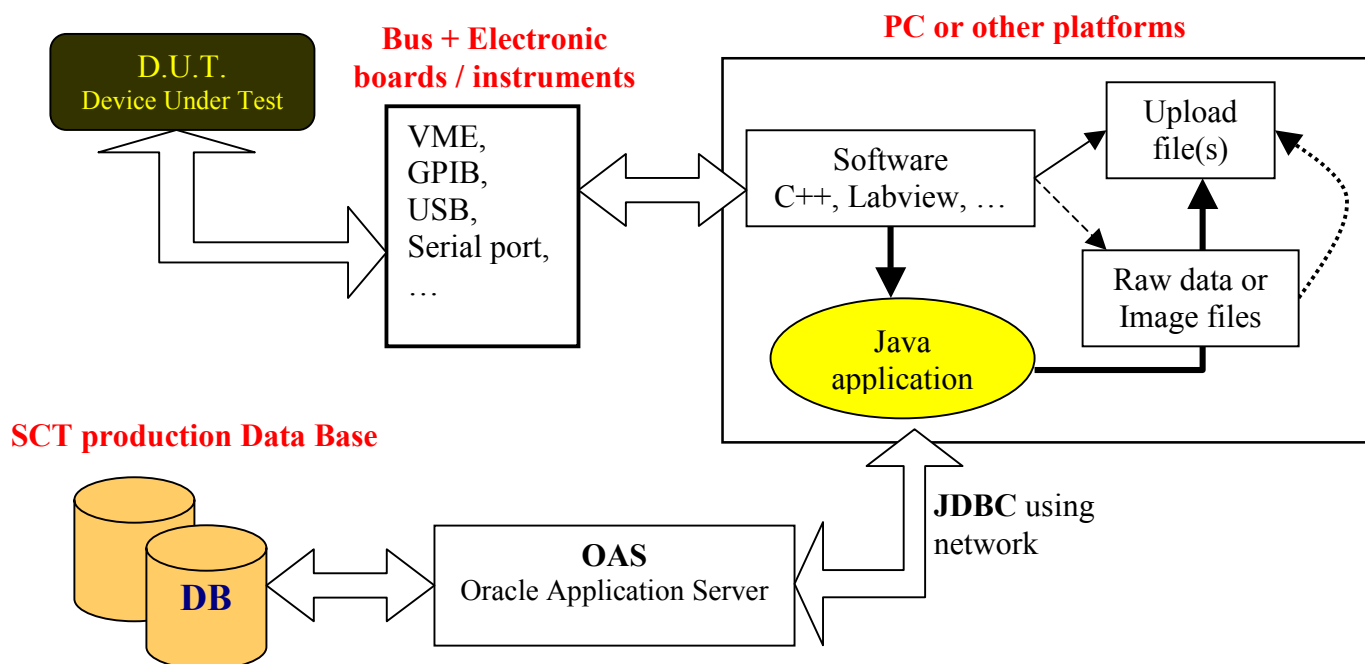


Figure 10: General scheme of a test system interfaced with the SCT production DB.

6. Conclusions

The SCT production database is working well since year 1999 and is based on an Oracle DB system located in Geneva University. The core data structure of this DB and its concept was frozen when the tools were distributed to the SCT users for recording the first production data. This DB was installed and set such that it is widely accessed from everywhere via the Internet and from any platform. The web access modules and the java applications are the basic tools that are distributed and maintained, which allow a large flexibility of inserting, consulting, deleting safely the data. Nevertheless the DB has a special read access channel allowing any external development tool for consulting or making DB reports.

Every SCT institute owns an access username and password ensuring the data property and reliability. One of the main advantages of this DB concept is that it is an evolutionary model allowing even if the core data structure is frozen, to extend later in time the test data structure. This was revealed to be very practical especially when the test procedures are not defined yet. Thanks to the java applications, mass data insertion and automation of acquisition system interfaced with the DB is possible.

References

- [1] The ATLAS Inner Detector Technical Design Report, CERN/LHCC/97-16 and CERN/LHCC/97-17 (1997).
- [2] A Database for Silicon Microstrip Detectors in the SCT Central Cluster Working Group, P. Demierre, D. Ferrère, ATL-INDET-98-219.

- [3] Proposal for a Database for the ATLAS SCT Construction, J. Foster, S. Snow, ATL-INDET-96-144.
- [4] Java Development Kit, <http://java.sun.com/products/jdk/1.1/>
- [5] ATLAS Part Identification, F. Ditus, ATC-OQ-QA-2040.
- [6] SCT welcome DB page, <http://melb.unige.ch:3143/>
- [7] Electrical tests of SCT hybrids and modules, http://hepwww.rl.ac.uk/atlas-sct/documents/Electrical_Tests.htm
- [8] Barcode and database entry of the series production of silicon microstrip detectors, http://jsdhp1.kek.jp/~unno/si_det/QC/BarcodeSCT.fm52.pdf

Nom du document : DBNote.doc
Dossier : D:\didier\Documents\Database\Publication
Modèle : C:\Documents and Settings\didier\Application
Data\Microsoft\Modèles\Normal.dot
Titre : Draft #1
Sujet :
Auteur : Didier Ferrere
Mots clés :
Commentaires :
Date de création : 02/04/2002 09:27
N° de révision : 129
Dernier enregist. le : 26/07/2002 16:09
Dernier enregistrement par : Didier Ferrere
Temps total d'édition : 2 816 Minutes
Dernière impression sur : 26/07/2002 16:10
Tel qu'à la dernière impression
Nombre de pages : 15
Nombre de mots : 4 328 (approx.)
Nombre de caractères : 22 076 (approx.)