EUROPEAN ORGANISATION FOR NUCLEAR RESEARCH

CERN – A&B DEPARTMENT

AB-Note-2008-012 ABP

BEAM SCRAPING FOR LHC INJECTION: HIGH LEVEL APPLICATION DEVELOPMENT

P. A. LETNES

Abstract

The Large Hadron Collider (LHC) at CERN (European Organization for Nuclear Research) will be the world's most powerful accelerator when it is commissioned during 2008. To operate the LHC, injection of very high intensity beams from the Super Proton Synchrotron (SPS) pre-accelerator is required. With intensities of more than 3 _ 1013 p=cycle, it is essential that there is virtually no beam halo present. Such particles can hit the LHC beam pipe, and may cause magnet quenches due to heating. Fast scrapers have been installed in the SPS to measure and remove any halo before the beam is extracted towards the LHC. Fast scrapers have been chosen because there is too little time available for beam cleaning with large collimators. The scraper hardware has been in place in the SPS ring for several years. A low level computer for controlling the scrapers is also in place. A high level control application was, however, not written at the time. The development of the missing high level control application is the subject of this work. The functional requirements for the application have been established in a report by G. Arduini and H. Burkhardt [2]. The application had to be written in Java to be compatible with the control system in the Cern Control Centre (CCC). Java is chosen due to portability between operating systems and due to the large number of freely available libraries. A working application for high level control has been developed and released into the software repository for the CCC computers. The application has been tested and used for two machine studies. Other users interested in using the scrapers for machine studies and commissioning have also tested the application. These tests indicate that the application works as expected, and can be used by operators in the future. Also, they have provided valuable feedback for further improvements of the application.

CERN, Geneva, Switzerland
14/02/2008

Preface

This report was written as a project report for my master's degree at NTNU. The work associated with the report was done during my stay at CERN as a technical student. CERN's technical student program is open to students having completed at least 18 months of technical undergraduate studies, excluding theoretical and experimental particle physics.

The goal of this report is to describe the development of the high level control application for the SPS beam scrapers, which was given as a technical student assignment. While not answering any of the open questions related to the physics of beam scraping, this report will describe the high level application in detail.

For CERN users, chapter 4 (describing the high level application architecture) and appendix 6 (a short user manual for the application) are probably the most useful chapters. See also `http://cern.ch/pletnes` for links to documents and web pages related to the scrapers.

This version of the report was adapted for publishing in CERN CDS, directed towards any CERN users. To obtain the version handed in to NTNU, please contact the department of physics at NTNU.

Contents

Chapter 1

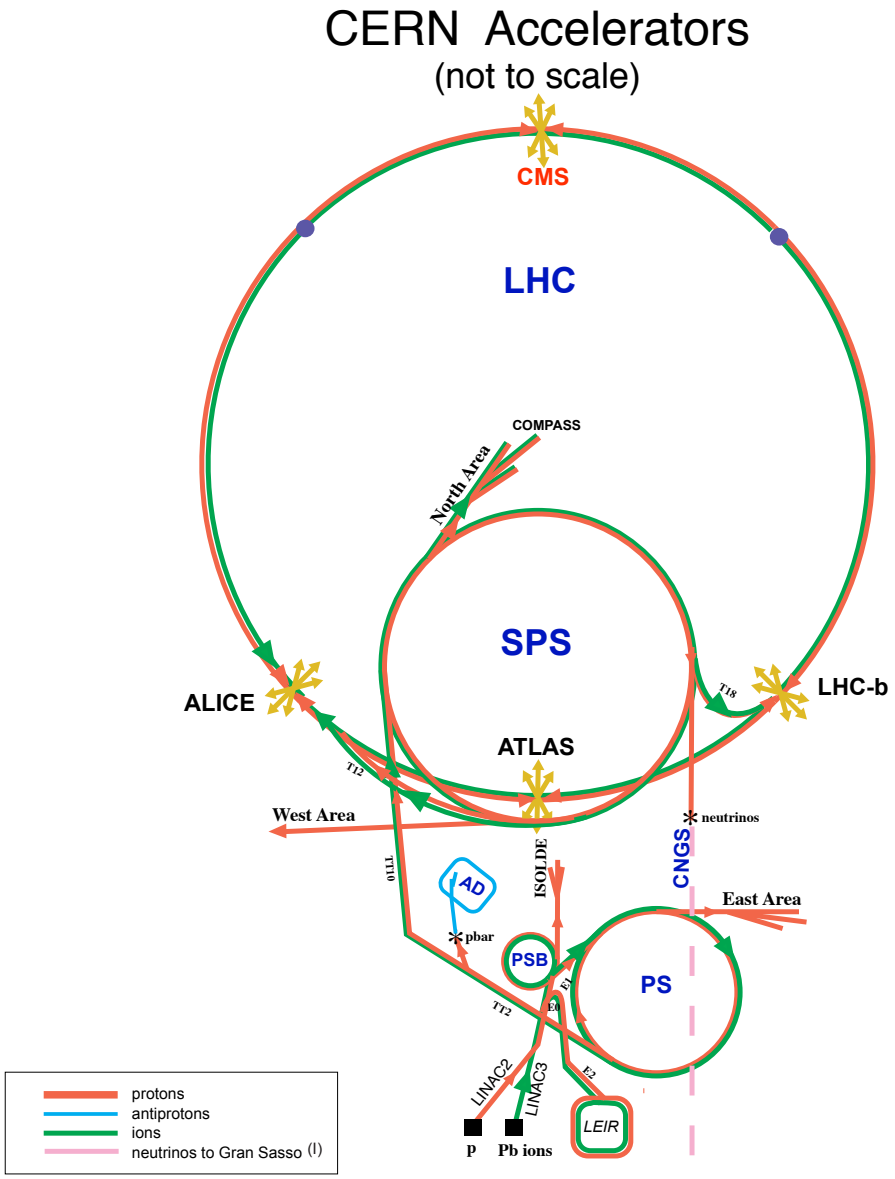
Introduction

This chapter will briefly explain the background for this work. The reasons for beam scraping are explained. A brief touch on the scraper hardware and its interaction with the beam is also to be found. Finally, a short repetition of the requirements for the control software shows the necessity of a high level application.

1.1 CERN and the accelerator complex

This work has been done at CERN (The European Organization for Nuclear Research), near Geneva, Switzerland. CERN is the biggest particle physics laboratory in the world, and is an international organization. To perform a wide range of experiments in fundamental particle physics, CERN utilizes a large complex of accelerators and colliders. The most recent and most powerful is the LHC, the Large Hadron Collider, with a centre of mass energy of 14 TeV and an unprecedented luminosity. For an overview of the accelerator complex, see Figure 1.1. The LHC is scheduled to be commissioned during 2008.

Rather than accelerating particles all the way from rest to 7 TeV in one machine, it is more efficient to perform the acceleration in several steps. As a rule of thumb, an energy gain of about 10 per machine is preferable. One important reason for this is that as the beam is accelerated, the transverse beam size shrinks in the lab frame. The beam shrinks because while longitudinal momentum increases during acceleration, transverse momentum is essentially conserved. Hence, the transverse momentum becomes relatively smaller at higher energies. In practice, this means that an accelerator needs a relatively big aperture for injection, while only a smaller aperture is necessary at top energy. It should be noted that aperture is an important cost



LHC: Large Hadron Collider
SPS: Super Proton Synchrotron
AD: Antiproton Decelerator
ISOLDE: Isotope Separator OnLine DEvice
PSB: Proton Synchrotron Booster
PS: Proton Synchrotron
LINAC: LINear ACcelerator
LEIR: Low Energy Ion Ring
CNGS: Cern Neutrinos to Gran Sasso

Figure 1.1: The CERN accelerator complex. Figure: R. Ley.

driver when building accelerators: Larger magnets cost more, consume more energy, and are in general more technically challenging.

The CERN philosophy is to use older accelerators as injectors for the newer, more powerful ones. For LHC experiments, this means that protons and ions start out in a linear accelerator, LINAC2 (protons) or LINAC3 (ions). In the case of protons, the beam is injected from LINAC2 into the PSB. After acceleration to a particle momentum of 2.1 GeV/c in the PSB, the beam is in turn injected into the PS.

The PS was the first large accelerator at CERN, commissioned in 1959. Since then, it has been used to accelerate antiprotons, protons, electrons, positrons and ions. Still in use, it delivers protons and ions to the AD, ISOLDE and SPS. For LHC injection, it accelerates protons to a momentum of 26 GeV/c.

After this, protons are transferred to the SPS via underground transfer lines. In use since 1976, the SPS has been used as a major collider machine, injector for the LEP and has supplied several fixed target experiments with particle beams. The SPS, being a 6.9 km synchrotron, is capable of accelerating the protons to a momentum of 450 GeV/c. Finally, the beam is delivered to the 27 km LHC, which stores and accelerates the beam to the collision momentum of 7 TeV/c.

Some of the most important components of the LHC are the superconducting bending magnets, the dipoles. The dipole magnets are very vulnerable to heating. If the magnet is heated above a certain temperature (depending on the desired field strength), the superconductor loses its superconducting property. This leads to electrical resistance, which in turns leads to more heating. This avalanche effect is called a magnet quench, from which it may take several hours to recover.

Extensive protection systems have been constructed to avoid both quenches and damage. The scrapers are a part of this protection scheme. Studies indicate that scraping at around 3.5 σ, where σ is the RMS beam size, will help prevent quenches during LHC injection while only reducing the beam intensity by 0.2% [3].

Due to aperture requirements at injection, regular collimators are too far from the beam axis to remove beam tails as small as 3.5 σ at top energy. The top energy plateau will only last on the order of one second. Figure 1.2 shows how an SPS cycle for LHC injection might look. Because collimators are big and heavy, they cannot be moved in and out again during this short time span. It has therefore been found that any transverse tails will have to be removed using fast scrapers [1].

It is very hard to remove 450 GeV/c protons in a single pass, as in a transfer line. In a synchrotron like the SPS, protons may pass through the

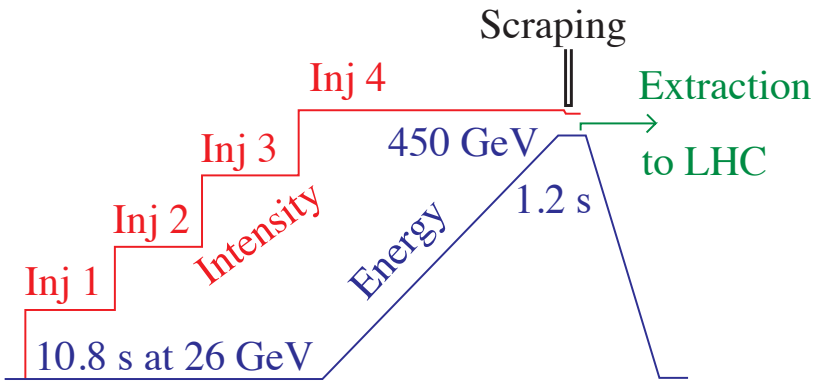


Figure 1.2: Schematic view of an SPS user cycle for LHC injection. First, four injections fill the SPS ring. Afterwards, the energy is ramped to 450 GeV. Note the 1.2 s plateau at top energy. Beam scraping will occur at the end of this plateau, immediately before extraction to LHC. Figure: H. Burkhardt.

scrapers several times, making it easier to remove the high energy protons. For this reason, the option of cleaning the beam in the ring has been preferred.

In addition to the quench issue, high amplitude protons will also contribute to the background noise in the experiments. It should also be noted that the contribution of these high amplitude protons to the collision luminosity is negligible. This indicates that scraping will improve beam quality in most if not all aspects.

1.2 Beam-scraper interaction

During scraping, the protons in the beam tail may pass through the scraper jaw one or several times. In each pass, the proton may encounter any of the following effects:

- Inelastic nuclear scattering, kicking out gamma photons and particles from the copper nuclei
- Multiple scattering, changing the direction of the proton's momentum
- Ionization, slowing the proton down and kicking out electrons and photons

An inelastic scattering is essentially a collision between a proton and a copper nucleus. It will remove the proton from the beam and usually generate secondary particles. Some of these secondaries will deposit their energy inside

the scraper jaw, while others will exit the scraper jaw and enter the beam pipe. These particles will not have the correct combination of momentum and charge/mass ratio to stay on orbit, and will hit a collimator or the beam pipe.

Multiple scattering is the process where a charged particle traversing a medium is deflected by many small-angle Coulomb scatters. The scattering is mainly caused by Coulomb interaction between protons and copper nuclei. Multiple scattering leads to a larger betatron oscillation amplitude, and causes the proton to be lost downstream.

Ionization is the process where protons kick out electrons from the electrons in the metal. This produces secondary electrons and photons, which also causes heating of the copper. Ionization will make the proton slow down slightly and possibly be lost in the momentum scrapers.

1.3 Scraper hardware

The scrapers consist of two "jaws", made of copper. There is one jaw and two electrical stepping motors per transverse plane (horizontal and vertical). The scraper jaws are intended to sweep quickly through the beam at top energy, right before the beam is extracted from the SPS and injected into the LHC. The scrapers are, as previously mentioned, not intended to absorb the particles themselves, but rather to scatter the protons into a collimator.

1.4 Control software requirements

The scraper hardware and servers running low level control software are in place. However, this control software can only be used by experts knowing the scraper system in detail. A more user-friendly, high level application is required for regular operation. Given the huge number of devices and instruments the accelerator operators need to control, it is essential that the user interface for each of them is as simple as possible.

The high level control software should basically be capable of changing settings in the low level server, and to give status and diagnostics information back. Some additional functionality is desired for convenience. Detailed functional specifications for the scrapers are found in [2]. This document contains a full requirement specification on all levels—software, hardware and physics. It is worth noting that some of these requirements are essential, while others are considered optional.

Chapter 2

Scraper hardware

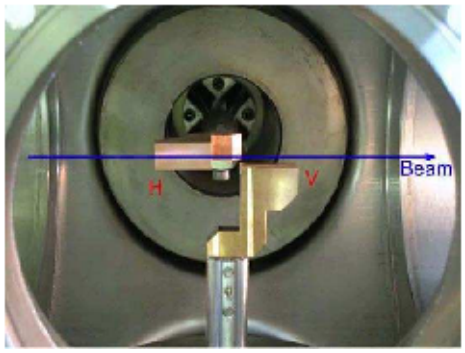


Figure 2.1: A picture of the scraper jaws.

The mechanical parts of the scrapers were originally used in the ISR collider. The essential parts are two stepping motors and one "jaw" for each transverse plane (horizontal and vertical).

While being more than 20 years old, the mechanics is robust and precise, and should be able to do its job for some time to come. However, it is expected that the existing scrapers will be replaced by a new system during the lifetime of the LHC. This is because they might not be compatible with high intensity operation in the SPS both for LHC injection and fixed target physics.

The current scrapers remove tails outside a rectangular "envelope". A diagonal scraper is needed to remove the sharp corners of the beam. This will reduce the beam intensity scraped, but still remove the unwanted tails. As machine aperture is a valuable asset, especially in superconducting colliders like the LHC, diagonal scrapers will most likely be installed at some point.

2.1 Scraper jaws

For a picture of the jaws, see figure 2.1. The scraper jaw itself consists of a prism of copper metal attached to two stepping motors. It is 30 mm thick in the direction of the beam, and roughly 10 mm thick in the directions transverse to the beam. Copper was chosen for several reasons: It has high electrical conductivity, is easy to work with in terms of mechanical properties, and has a relatively high melting point (1085 °C). Copper has 29 protons in its nucleus. This is a nice intermediate number; heavy enough to give significant stopping power, while light enough to let most protons pass through.

The SPS beam for LHC injection contains considerable energy, and could damage the scraper in case of a full impact. Thermomechanical studies should be performed to assess the damage potential and establish limits for safe use.

2.2 Motors

There are two electrical stepping motors for each scraper jaw. One motor is fast moving, with a speed of 20 cm/s. The high speed is chosen because there is, in the most challenging user cycles, less than one second available for scraping. This motor only moves between two physical buttons. It has no precision position measurement: It just stays at the one end switch until it is told to move, then moves all the way to the other end switch.

The other motor is a precise, slow moving stepping motor. A resolver measures the jaw position associated with this motor. It is this motor that determines the position of the scraping. The slow moving motor will move the scraper jaw to the scraping position, and then the fast moving motor will quickly sweep the scrapers through the beam. The scraping position can be controlled with a resolution down to 10 μm [5]. For comparison, the RMS beam size is about 0.6 mm at the scraping point. For a detailed discussion about the scraper movement, see section 3.3.

The slow moving motors are equipped with "retract" end switches. The end switches for the slow motors are, as for the fast motors, physical switches that are touched when the scraper is fully retracted. These end switches are not used for the scraping itself, but are instead a safety measure. If the scrapers leave the slow end switch, an interlock will be set. This will in turn cause the SPS beam to be dumped unless the interlock has been manually masked (or "ignored") by the SPS operators on duty. Any device capable of moving into the beam has an interlock associated with it, to prevent damage

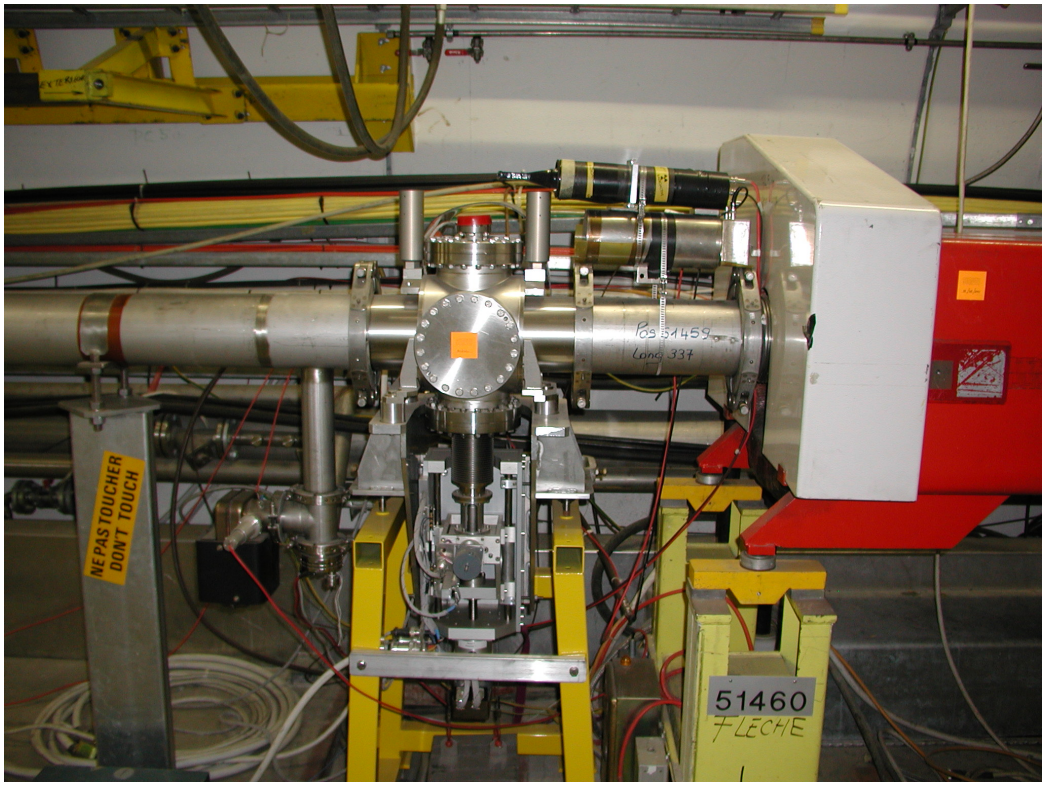


Figure 2.2: The scrapers pictured in the beam pipe.

to components inside the accelerator.

2.3 Associated collimators

Associated with the scrapers are a primary and a secondary collimator. These are designed and placed to absorb the protons scattered in the scrapers. The collimators are placed, respectively, after roughly 90° and 180° phase advance. The idea is that while the scrapers only scatter the particles slightly, the collimators are made to slow down and absorb the particles.

Each collimator consists of four collimator blocks: Two blocks for each transverse plane. These blocks are installed symmetrically relative to the beam axis, but can be positioned independently. Each block in the primary collimator consists of a 100 mm long tungsten core, contained in two copper end pieces. The end pieces are made from copper to reduce RF losses and to extract heat from the tungsten core. The block has a total length of 450 mm and can be positioned with a resolution of 5 μm. The blocks for the secondary collimators are similar, but their length is only 250 mm due to the fact that they have shorter copper end pieces.

Chapter 3

Low level control

This chapter intends to give an overview of the low level control system. It also explains how the settings made in the high level application are translated into movement of the scraper jaws.

3.1 Overview of the low level control

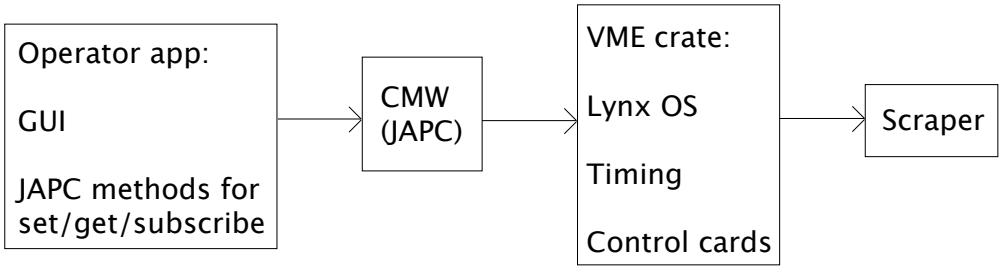


Figure 3.1: Software architecture overview. The VME crate is the low level control server. To communicate with the VME crate, the high level application uses JAPC, which acts as Control MiddleWare.

As mentioned earlier, the hardware is installed in the SPS ring (at position 51659). The hardware itself is connected to a VME crate. The VME crate contains cards for communicating with the physical hardware, which in this case consists of four stepping motors. The crate is also connected to the accelerator timing system, to make sure the scraping is conducted at the correct time.

The VME crate runs the operating system LynxOS. On this OS, the FESA server (or FESA class) is run. The VME crate is thus the FEC (Front

End Computer) in this case. The FESA server is responsible for controlling the four motors via the control cards. The movement cycle specified in the FESA server is the only way of using the scrapers. Only some position and timing settings for this movement cycle can be changed. The movement cycle is discussed in detail in section 3.3.

FESA is a framework for standardizing instrument control. The FESA framework can be run on different computers and operating systems (Lynx, Linux or Windows), but these details are hidden from the user and the GUI developer. To quote the FESA home page (`http://project-fesa.web.cern.ch/project-fesa/`):

> FESA is a comprehensive framework whereby front-end software is to be designed, developed, deployed and maintained according to the AB standard.

On this web page, one can also launch the FESA shell, which lets you examine and edit settings directly to the FESA server. While possible, it is not convenient to use the FESA shell directly to control the scrapers: The FESA class contains a lot of technicalities which makes it unusable for everyone but the experts knowing the instrument intimately.

For regular operations, a high level control application had to be written in Java. This application uses CMW (Control MiddleWare) to communicate with the FEC. In this case, the CMW is the JAPC framework, which hides all the FESA details behind Java classes and methods. The *parameter* abstraction, which typically represents a control value of an accelerator device, is the central abstraction in JAPC. Using this abstraction, JAPC supplies get, set and subscribe methods for communicating with the FEC. These methods are used to get information about the device and to make settings to it. The diversity of different devices, programming languages and so on is handled "under the hood": Each type of device needs to have its own implementation of JAPC, but from the GUI programmer's perspective, you can treat them in the same way.

3.2 The FESA class

The scrapers are used by saving certain settings to the FESA server. Each setting is saved in a field, which can be a string, a number, an array of numbers and so on. With this information in hand, the FESA server takes appropriate action at the appropriate time: move stepping motors, wait for events in the user cycle, publish status updates, and so on.

There are many expert setting fields in the FESA class, like the speed of movement of the scraper jaws. These are set by an expert after MDs, and normally they should not be touched later. However, there are a few fields that are meant to be changed on a regular basis:

type The scraping type. Can be set to Horizontal plane, Vertical plane, Both planes, or Idle.

hPosition in millimeter. The horizontal scraping position. Given in the "absolute" coordinate system with $(0, 0)$ as the nominal beam center.

vPosition in millimeter. The vertical scraping position.

outTargetDiff in millimeters. The distance the scrapers move out from the **hPosition** or **vPosition** (from point "2" to point "3") after scraping, see figure 3.2. Given as a difference relative to the scraping positions, hPosition and vPosition.

parkTargetDiff in millimeter. The length from the scraping position (point "1") to the parking position (point "0/5"), see figure 3.2. Given as a difference relative to the scraping positions in the same way as outTargetDiff.

delay in millisecond. The time delay from the beam injection event to the time of scraping.

The use of these settings is explained in section 3.3.

In addition to these six fields, which can be regarded as input, there are also a large number of output fields. The FESA server publishes status and results from scraping in these fields. Only some of these fields are interesting for regular operations. The most important fields are:

- Which user cycle the real time status and results were received from
- The last received super cycle number
- A message indicating if the last scraping went fine
- Scraper status: OK or not
- Status of the end switches (the buttons monitored by the interlock system)

Both status fields (feedback from the scraper) and setting fields (instructions to the scraper) in the FESA class are organized into so-called FESA properties. These properties make the programming more organized and defines the communication interface between the FESA server and the high level application. While there are many FESA properties, several of these contain fields for internal use only. The five FESA properties used by high level control are:

RetractAll For retracting the scrapers

Setting Contain all position and timing settings, common for all user cycles

Control Contains only the scraping type, the only setting that is user cycle dependent

Status Reports on the status of the scrapers: End switch status, error messages and so on

ScrapRt Contains user cycle dependent results of scraping: Super cycle number, a message stating the result of the scraping, and several other fields

Needless to say, only the information useful for normal operation should be displayed on screen. The rest can optionally be saved to a file for offline analysis.

3.3 Scraper movement and timing

Figure 3.2 shows how the scraper moves in the horizontal plane. The scraper in the vertical plane works in exactly the same way as the horizontal scraper, except for the fact that it is rotated by 90° around the beam axis relative to the horizontal scrapers. It should be noted that the settings parking position, out position, and delay are shared for the two scrapers, while scraping position is set individually for the two scraper jaws.

The movement of the horizontal scraper relative to the SPS beam is shown in figure 3.2. The scraper starts out parked at point "0". (This is also the parking position of the scraper if **type** is set to idle.) The scraper is here all the way down, touching the lower "fast" end switch. The horizontal position is set to the scraping position plus the relative distance **parkTargetDiff**.

After beam injection, the scraper is first moved left to the scraping position, point "1". The timing of this is calculated by the FESA server, depending on the value of the **delay** field. This movement is done with a slow but precise stepping motor. The motor stepping resolution is 10 μm.

The FESA server will now wait until the time for scraping has been reached. Then, at the correct time, the scrapers will move quickly upwards through the beam until they reach the upper "fast" end switch, point "2". The scrapers sweep through the beam with a velocity of 20 cm/s. Depending on the beam size, the scrapers will spend on the order of 10 ms−100 ms in the beam, performing the actual scraping. For comparison, MAD-X calculations predict the RMS beam sizes to be $\sigma_x = 0.62$ mm and $\sigma_y = 0.58$ mm at the point of scraping (LHC nominal beam numbers). The actual time dependence of scraping is hard to specify, as it depends on how the machine is set up: Scrapers, downstream collimators, machine parameters and beam sizes all have an impact here.

After this, the scraper is moved slowly out to point "3". The distance between "2" and "3" is specified in the FESA field **outTargetDiff**. Finally, the scraper completes the movement cycle by moving quickly down to point "4", then slowly in to point "5". The scraper is now ready for a new user cycle. It should be noted that the distance from points "2" to "3" can be set to any nonnegative value, thus it is possible to scrape twice at the same position if the **outTargetDiff** is set to 0.

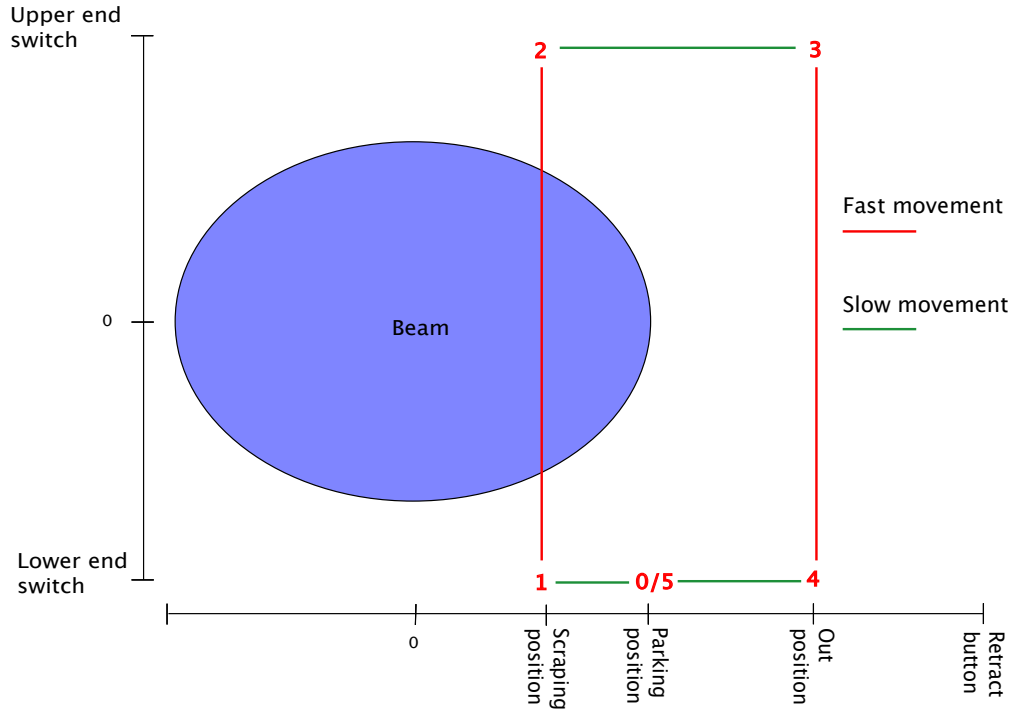


Figure 3.2: The movement of the scrapers relative to the SPS beam. Note that the beam size is greatly exaggerated.

Chapter 4

Scraper control application

To use an instrument or a device for the accelerators in regular operations, a high level control application is needed. This application is an important step towards making the scrapers a fully operational tool for SPS operations during LHC injection. A brief user manual for the application is found in appendix 6.

4.1 Software requirements and users

The original requirements for the scraper application are stated in [2]. This document also contains requirements for beam physics and hardware, both in regular operations and for MDs. Some of the requirements are considered essential, while others are optional, and will be implemented if time and resources allow.

First and foremost, the application must be able to control the position and timing settings. These include the scraping position for each transversal plane, scraping time, out position and parking position. It must also be possible to fully retract the scrapers at any time. It is desirable that the application is able to save settings and status updates to file, for later analysis or for restoring settings to previous values.

The application should also be able to make predefined settings to the associated collimators, like "collimators in" and "collimators out". It may also be convenient to include monitoring of certain instruments in the scraper application, to be able to quickly see the effect of the scraping on the beam.

The main users of the scraper control application will be the SPS operators. The operators are responsible for the daily running of the accelerators, and for delivering beams with different properties to different experiments. To make the scrapers operational, it is essential that the scrapers have an

easy-to-use interface with behavior similar to other accelerator control applications.

In addition, it is assumed that some physicists will be interested in using the scrapers as an MD tool. It is an advantage if the scrapers can be used by an operator or the physicist directly while performing MDs, without needing an expert to control the scrapers (which has been the only option until recently).

4.2 Development environment

At CERN, all accelerator control software is written in Java. Java is chosen for its ease of use and rich library (called Java Swing) of graphical components freely available. Java is also platform independent, meaning that the bytecode compiled application runs equally well on the Linux computers in the control room and on Windows terminal servers used by developers.

While Java is not always as efficient as other programming languages, it is "developer friendly" to make graphical applications with Java. The advantages of having, for instance, networking capabilities built into the standard libraries, outweigh the disadvantages of using some extra computer memory. Not having to develop already existing functionality saves a lot of time and money.

Java Swing is a widget toolkit for Java. It is a platform independent toolkit, making applications behave and run very similarly on any platform with a Java Virtual Machine (JVM) implementation. Swing supplies so-called "lightweight" graphical components. Lightweight is here defined as behaving in the same way on all JVMs, and written entirely in Java.

To communicate with lower level software, the JAPC framework is used. The JAPC framework supplies Java classes and interfaces, hiding implementation details of the low level control software from the graphical application developers. This gives a nice decoupling between the low-level control of devices and the development of graphical control systems. For more information on JAPC, see `http://controls-wiki.web.cern.ch/controls-wiki/japc/`.

When a version of the application is complete, it is released from the accelerator software CVS repository into a software repository accessible from the CCC computers. The applications in this repository are available in at least three versions: Old, Production, and Developer. The production version is normally the latest stable version. If one wants to add and test new features, one can make a developer release. The software repository is available as a web page, which you can browse in any web browser. Appli-

cations are launched by downloading and running a .jnlp file, which contain the information necessary to download and run the application itself.

4.3 Implementation details

Some readers may not be familiar with Java or Java Swing. For an easily accessible discussion about Java, Java Swing, events, listeners, and so on, see [6].

The most important classes in the application are shown as a UML diagram in figure 4.1. The Scraper class has several important tasks, and acts as a mediator between the GUI and the FESA server. The `DeviceManager` class is used to manage the actual communication with the FESA server. The two classes extending `JTabbedPane` are the second-to-top level components in the graphical hierarchy (the `Scraper` class being the top level component). Finally, all four relevant FESA properties, `Status`, `ScrapRt`, `Control`, and `Setting`, have a Java class representation. These Java classes extend the `AbstractProperty` class.

The classes extending `AbstractProperty` are containers for the fields in the various FESA properties. For instance, when the `Scraper` class wants to make a setting to the FESA server, it gets a new `Control` and a new `Setting` object from the GUI. Afterwards, these objects are passed to the `setValue` method in `DeviceManager`, which reads the fields from the object and makes the setting to the FESA server. `AbstractProperty` objects are used in a similar way when a status update is received from the FESA server.

The classes derived from `JTabbedPane` contain the rest of the graphical component hierarchy: panels, buttons and so on are created and updated here. In the `StatusTabbedPane`, `setStatus` (and similar methods) are called when a new value is published from FESA. These methods then update the text fields and other graphical components contained in this tabbed pane. In the `ScraperTabbedPane`, the `getSetting` and `getControl` methods construct, respectively, a `Setting` and a `Control` object from the input fields shown on screen.

The `DeviceManager` is created and initalized during creation of the `Scraper` object. Later, it is used to handle communications with FESA. The `getValue` method is used to initialize the input text fields in the GUI with the current values from FESA. The `setValue` does exactly the opposite; make settings from the GUI to FESA. The two methods `startMonitoring` and `stopMonitoring`, respectively, start and stop subscriptions.

Starting subscriptions works by adding a `ParameterValueListener` (here: The `Scraper` object) to a list of listeners. Every time a status update is

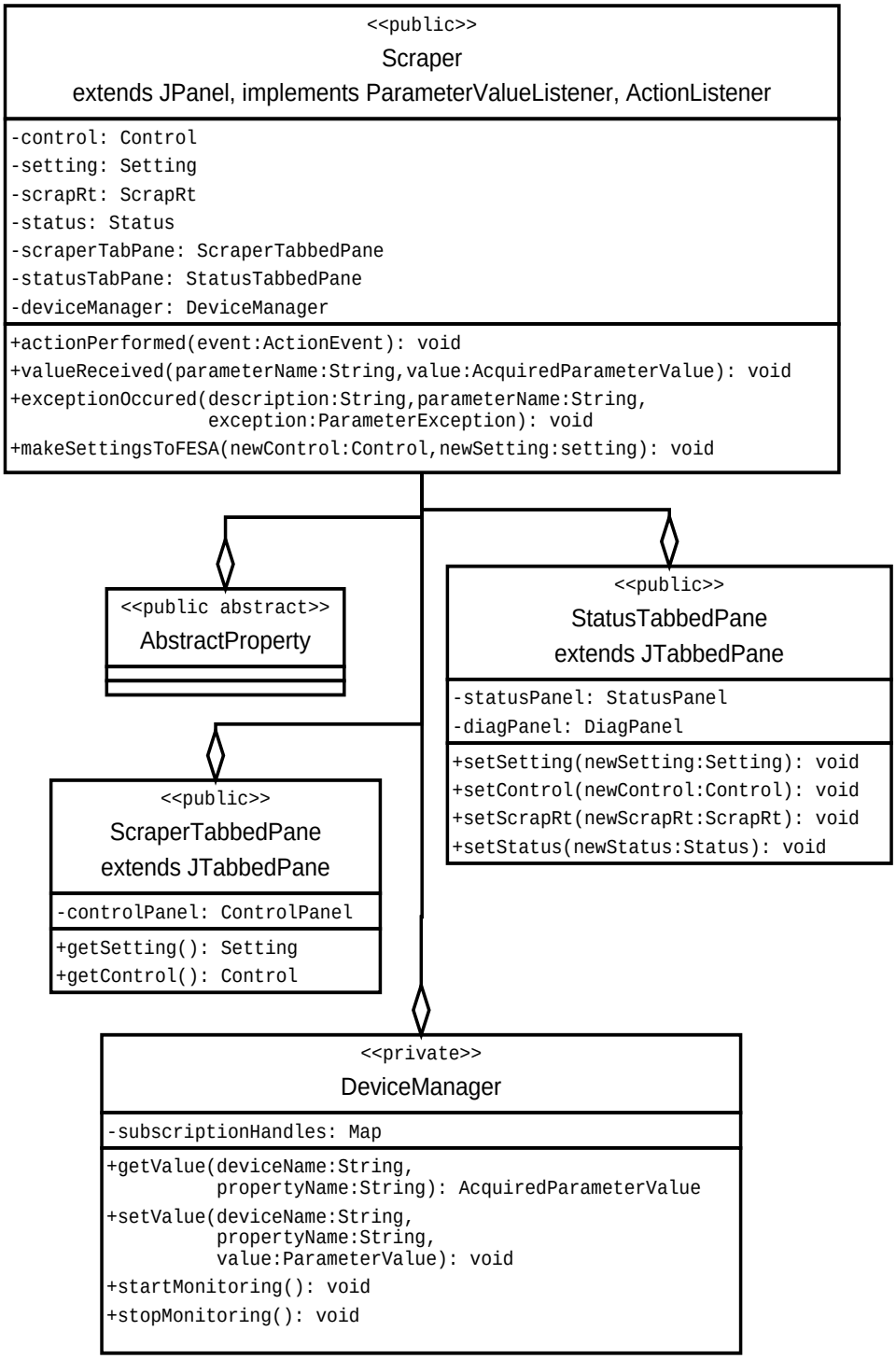


Figure 4.1: UML class diagram of scraper application. Only the classes and members discussed here are included.

published, the `valueReceived` method (specified by the `ParameterValueListener` interface) in each listener is then called. In the `valueReceived` method, the `Scraper` object then updates all the graphical components with the data received from the FESA server.

The `Scraper` class, being the mediator, contains references to all other classes, directly or indirectly. This means that the `valueReceived` method can call, for instance, the `setScrapRt` method in the `StatusTabbedPane` class, which then updates the status information shown on screen. In a similar way, the `actionPerformed` method can call `setValue` in the `DeviceManager` class, and so on. The `actionPerformed` method is specified by the `ActionListener` interface, and is called every time the user interacts with the graphical interface—for instance, clicks a button with the mouse. This is how the Scraper class connects the other parts of the application.

4.4 Constraints imposed by the application

The scraper control system has been developed with scraping during only one user cycle per super cycle in mind. All scraping settings, including timing, are shared amongst all user cycles. This reduces the flexibility of scraping during several user cycles a great deal. Also, the high level application is designed to set scraping to idle when changing user cycles. The application has been designed to behave like this to avoid memory effects. Assume that the FESA server has been set to scrape during a specific user cycle. If this user cycle is not used for a while, it will be undetectable that the scrapers are set to scrape. However, when this user cycle starts up again, the FESA server will resume scraping, most likely at an unfortunate time. Therefore, the application has been set to retract scrapers when it exits, and when the application user changes user cycle.

The time requirement is also important: It takes time for the scrapers to perform one movement cycle. It has been established that 10 s is the minimum length for a user cycle, if scraping is to work properly. Also, it is possible to receive real time status (**ScrapRt**) from the wrong user cycle if scraping occurs too close to the end of the user cycle. This is because it takes time to move the scrapers out of the beam.

Some of these constraints were discovered during software tests, see chapter 5.

Chapter 5

Machine studies and software tests

This chapter contains a brief report on software tests and machine studies conducted with the scrapers. Previous MDs have investigated scraping efficiency and tail repopulation. With the high level application, time dependence of scraping and tail repopulation experiments have been conducted.

5.1 Previous MDs and tests of the hardware

Several MDs and tests of the hardware have been conducted, proving both that the mechanical system is working well, and that scraping is done efficiently.

The main issue regarding scraping of the beam is that the high amplitude particles (also known as (non-gaussian) tails) are removed efficiently, and that the tails are not repopulated too fast. If the scraping does not remove the tails, the whole effort is of course wasted. Also, if the beam tails repopulate very quickly, that is, with a timescale shorter than the order of one second, there is no point in scraping. Tail repopulation is the process where particles with a small betatron amplitude get a large betatron amplitude. This phenomenon can be caused by several physical processes.

A study by M. Facchini et al. [5] indicates that scraping is efficient and that the repopulation time is longer than the one second required for LHC extraction. These results were obtained with the SPS working in the same pulsed mode that it is expected to operate in during LHC injection. The effectiveness of the collimation was also investigated and found to be satisfactory: Beam losses were only observed in the two collimators associated with the scrapers. No significant losses due to scraping were observed else-

where in the ring. One issue that was not investigated thoroughly here, is the issue of neutral particles: The collimators are designed to absorb charged particles like protons, and are not designed to absorb neutral particles like photons and neutrons generated by the protons colliding with copper nuclei in the scraper jaw.

A study by H. Burkhardt et al. [4] show that with a coasting beam (the beam is circulating with a constant energy of 270 GeV, but kept bunched by the RF systems), the repopulation time scale is on the order of five minutes. This study also concluded that even with the collimators fully retracted, most of the beam losses were concentrated close to the scrapers.

5.2 Tests of the high level control application

The high level control application has been tested at a number of occasions. Prior to the first real MD, an SPS machine stop (with no beam present) was used to check that communication with the low level software was working. A number of small bugs in the low level software were found. These tests also resulted in some ideas for diagnostics and status messages necessary to implement in the high level application.

Afterwards, two real MDs have been performed. Especially for the first MD, which was relatively short, it was expected that there would be little time for real physics experiments. In the end, however, both MDs were used to examine tail repopulation. These experiments showed that the high level control application was capable of controlling the scrapers as desired.

Software tests

During the final software test without beam, scraping was attempted during a short (4 s) user cycle used for parallel MDs. A parallel MD is when some user cycles deliver beam to physics experiments, while a short MD user cycle is available for machine experiments. This model disallows, for instance, coasting beams for MDs. It was discovered that this user cycle was too short for the scrapers, in the sense that the status messages were received during a different user cycle. This turned out to be very confusing. It was found that 10 s was the minimum user cycle length [7]. Also, scraping cannot occur too close to the end of the user cycle. This is because the FESA server needs a certain time to gather and publish status information after the scraping. Unfortunately, the exact length of time needed is not know at the time of this writing.

MD 1, September 12, 2007

The first MD was primarily intended to be a short (2 h) demonstration of ability to scrape the beam as desired. Using the low resolution BCT (Beam Current Transformer) in the SPS, the beam intensity before and after scraping was measured. It was evident that the scrapers worked, and that the timing of the scraping was according to settings.

After showing that scraping worked as hoped, the user cycle was changed to coast. In this user cycle, the particles in the beam are kept bunched by the RF systems, but with constant energy. The beam is not dumped and injected each super cycle as normal. The beam is instead kept circulating for as long as desired.

To measure tail repopulation, one first makes an initial scraping at a given point. After waiting for either (roughly) 5 min or 30 s, one scrapes again at the same position to detect any new tail particles. When particles touch the scraper jaw, secondary particles are emitted, showing up as beam losses in BLMs (Beam Loss Monitors). In this way, the scraper is used both as an instrument for removing tails and detecting them. Scrapers combined with BLMs is actually one of the most sensitive methods for detecting beam tails, which are frequently less well understood than the parts of the beam close to the beam center. While repopulation was detected, the time constant was on the order of minutes. This means that scraping is efficient for LHC injection, given that the flat energy top at 450 GeV only lasts less than 1.5 s.

MD 2, September 25, 2007

In the first MD, machine setup was relatively "quick and dirty". For the second MD, which lasted for 4 h, more time was spent on doing a careful setup. The results in the second MD diverged somewhat from those in the first MD: While MD 1 showed similar (and significant) tail repopulation in both planes on a 5 min time scale, MD 2 showed tail repopulation only in the horizontal plane. No model exists to explain this discrepancy. Two hypotheses have been proposed: One is that some of the particles in the beam may experience resonance effects, assuming nonzero chromaticity. The other one is that intrabeam Coulomb scattering kicks some particles into the tails. These hypotheses have however not yet been studied in detail.

During the second MD, a group working on BLMs also conducted a series of measurements. Amongst other things, the time dependence of beam loss was investigated. These measurements can be used as a check of simulations later: If a numerical model can recreate the same time dependence, one will be more confident that the model is a good representation of reality.

Both MDs resulted in several interesting measurement series. These include BCT intensities and BLM loss distributions (both in time and space). A full analysis of the data sets remains to be done.

Chapter 6

Scraper application v. 1.0.0 user manual

This section is devoted to explaining how the scraper control application version 1.0.0 looks and works. (Newer versions are being developed with the possibility of scraping during several user cycles in the same super cycle. These will have version numbers higher than 2.0.0.) The scraper application can be started from `http://abwww/ap/dist/sps/sps-app-scraper/1.0.0/` by running the `SPS-Scraper.jnlp` file.

A screenshot of the application is given in figure 6.1, giving an example as to what the application may look like while running.

The Console

In the Console, the program prints subscription updates, warnings and error messages. For instance, the Console will print a small message each time a status update is received. This serves to show that the subscriptions and application is working and running. Another example is if there is some problem with the status subscriptions: This will generate a Java exception, which prints an error message to the Console.

The Scraper control tab

Inside this tab, one has all the controls needed to make settings to the FESA server. The **Scraping type** box lets the user choose to scrape in the horizontal plane, the vertical plane, or both planes. This box also offers the "idle" option, meaning that the scrapers park at the parking position, but do not fully retract. This makes it faster to resume scraping again at a later time.

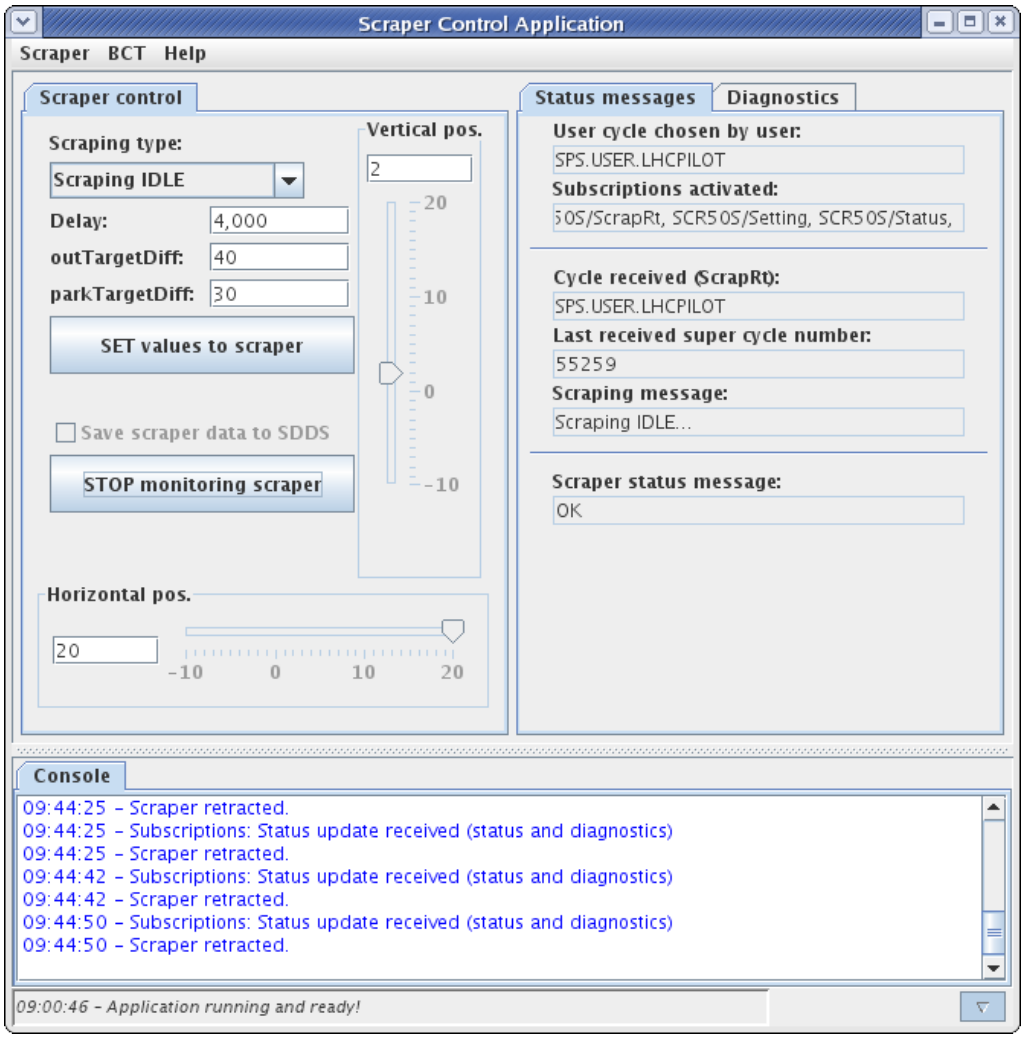


Figure 6.1: Scraper control application screenshot. Note that the application has booted, but no changes have been made, and no status subscriptions have started.

The **Vertical pos.** and **Horizontal pos.** fields let the user select the position of the scraping in terms of millimeters from the nominal beam position (which is 0). The positions are of course set independently for both transverse planes.

The **Delay** field lets the user choose the time of scraping in terms of milliseconds after beam injection. This setting is the same for both transversal planes.

The **outTargetDiff** and **parkTargetDiff** let the user set the outTargetDiff and parkTargetDiff positions, described in chapter 3. Both positions are given in terms of millimeters from the point of the scraping. These settings are the same for both transversal planes.

The **SET values to scraper** sets all six control values (Scraping type, delay, outTargetDiff, parkTargetDiff, vertical targetPos and horizontal targetPos) to the FESA server.

The **START/STOP monitoring scraper** will let the user start or stop subscriptions to status updates from the scrapers. If the **Save scraper data to SDDS** box is checked, the application will save one SDDS file for each published update. The saved data is usually not interesting, but the data can be used to examine what the FESA field values were at a given time. This can help an expert debug the FESA server or the high level application.

The Status messages tab

This tab contains feedback from the FESA server and the application itself. **User cycle chosen by user** is a reminder, telling the user of the application which user cycle he/she has chosen to work in.

Subscriptions activated indicates which of the FESA properties (Control, Setting, ScrapRt, and Status) are being monitored. If a subscription has not been successfully activated, the corresponding FESA property name will not show up here.

Cycle received shows the name of the user cycle received from the subscription to the ScrapRt property. If this name does not correspond to the name chosen by the application user, the FESA server is not publishing the status from the correct user cycles. This problem is to be expected if the user cycle is short, in the sense that the scraper does not have time to move out from the scraping in the same user cycle as it scraped. If the user cycle is long enough, a discrepancy here indicates a problem with the FESA server.

Last received super cycle number is from the ScrapRt subscription. This updates only if scraping is activated, and shows which super cycle the current status messages are from.

Scraping message shows a short message indicating if scraping has occurred and if it was successful (the best you can hope for here is "scraping went fine, waiting for next").

Scraper status message shows a message indicating if the scraper is fine. This field can have one of four values: Error, warning, OK or unknown.

The Diagnostics tab

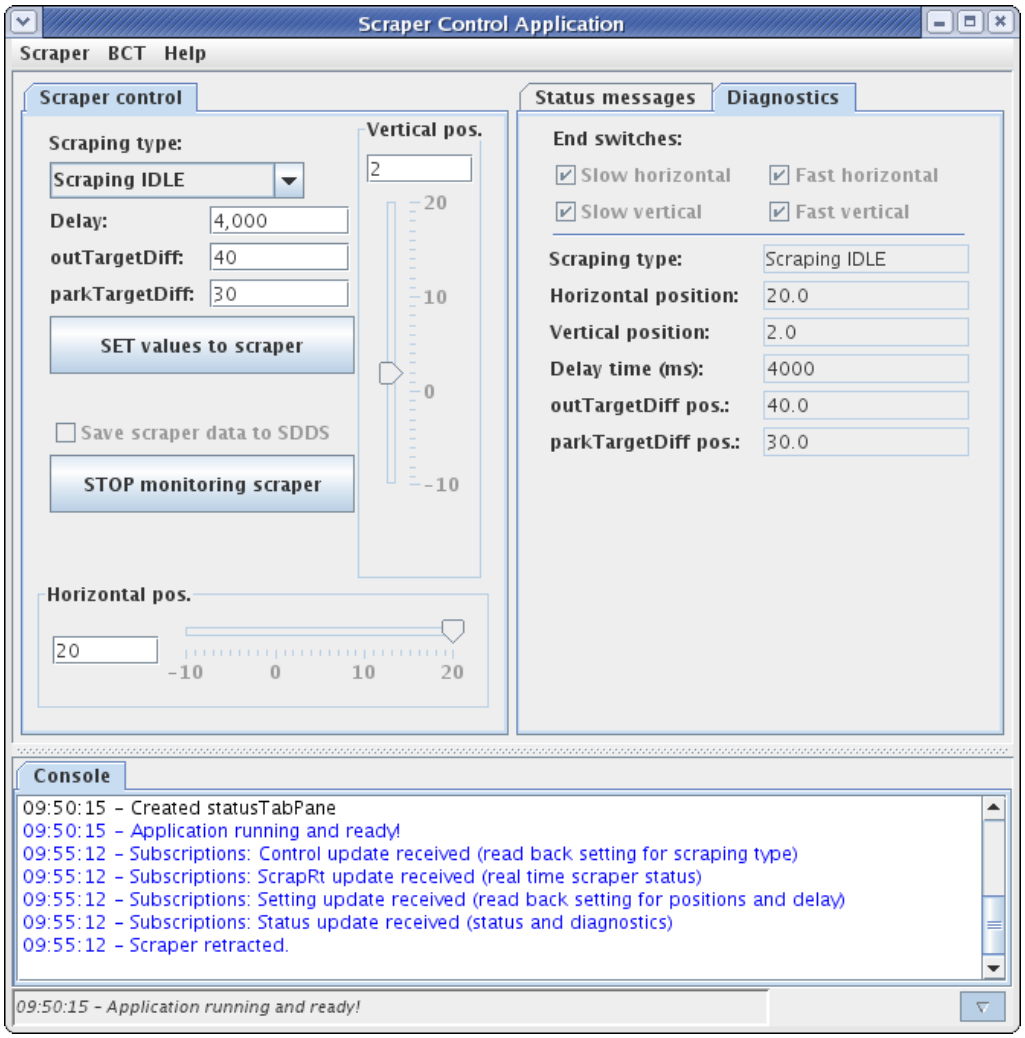


Figure 6.2: Scraper control application screenshot. In this screenshot, the Diagnostics tab is displayed.

This tab is only meant to be used as a tool to find out if the FESA server

and the high level application are working properly. A screenshot of the diagnostics panel is given in figure 6.2.

The **end switches** check boxes shows which of the end switches are pressed. These end switches are the same ones monitored by the interlock system. If all end switches are pressed, the scrapers have been successfully retracted.

The six fields under the end switch boxes are read-backs of the settings made to the FESA server. If the application user has changed, for example, the delay, he/she can see here if the server accepted the settings. These settings are published only when the application user changes the settings on the server by pressing "SET values to scraper".

The menu bar options

The menu bar contains some functionality that the user will not need to look at or use very frequently.

The **Scraper** menu contains three menu choices:

Choose cycle... Opens a dialog box which lets you choose the user cycle to work with.

Retract scrapers Set scraping to idle, and retract scrapers to slow end switches.

Exit Exits application.

The **BCT** menu contains two choices: High- or low resolution BCT plots. These menu choices create separate windows with BCT plots, allowing them to be inspected while controlling the scraper. The plots are simple and are only meant to give a quick overview. For instance, one can see the time of the scraping as a fast reduction of the beam intensity. The two BCTs available are one with high resolution but a low saturation limit, and another with a lower resolution but a very high saturation limit.

The **Help** menu contains some self-explanatory menu choices meant to give the application user some hints about how to use the scrapers.

Chapter 7

Summary and conclusion

The high level control application for the scrapers is now ready to be tested by SPS operators. The feedback received so far indicates that it is already usable for operations. Tests and machine developments indicate that the application is working as it should, but have demonstrated some constraints by the hardware and low-level control system.

For normal settings to the scraper, a user cycle should last at least 10 s. Shorter user cycles may result in incorrect behavior. The exact time requirement is determined by adjusting the position settings. Also, scraping very close to the end in a cycle can result in incorrect status messages from the FESA server. It may be possible to relax this constraint somewhat by fine tuning the movement speed of the scraper.

The FESA server also saves only one copy of the scraping delay field. This means that scraping must occur at the same time in the user cycle, should it be desirable to scrape during different user cycles in one and same super cycle. However, the current version of the high level application only allows scraping in one user cycle per super cycle.

MDs have shown scraping to be efficient. Tail repopulation has been shown to be small on one-second time scales, which is the requirement for LHC injection. While leaving some effects not understood, there is considerable empirical evidence to support scraping as an efficient way of removing beam halo.

7.1 Future work

While the work on a first version of the scraper control application is largely complete, there are still some remaining physics issues. It is not clear whether the scrapers can survive the full impact of nominal LHC beam intensities,

should the scrapers be moved too far into the beam by accident. Also, it has not been investigated where protons and secondary particles go after scraping. Scraping both with [5] and without [4] collimators show only local beam losses on the BLMs. It would be useful to investigate this more closely, to understand why these collimators have so little effect on beam losses.

In addition, the issue of replacement scrapers has not been completely settled. While a dedicated system may be installed in the future, other issues currently have a higher priority. Aperture requirements will probably force installation of diagonal scrapers at a later time. Only when one is approaching nominal LHC intensities will all these issues be settled.

Acknowledgements

First and foremost, I would like to thank Helmut Burkhardt for being a dedicated, inspiring and helpful supervisor. Always busy, but always has time for students.

I received vital help with the high level application from Jörg Wenninger and Grzegorz Kruk, their help is warmly acknowledged.

The hardware and low level software chapters own a lot to Stephane Bart Pedersen. I would also like to thank Arnaud Brielmann for interesting MDs and some araldit.

I would like to thank Hubert, Simon and Yngve for helpful coffee discussions about computer problems, beam physics and life in general.

Bibliography

[1] M. Benedikt et al. LHC Design Report, volume 3: the LHC Injector Chain. Technical report, CERN, 2004.

[2] H. Burkhardt and G. Arduini. Transverse tail scraping in the SPS for clean LHC injection. Scraper functional specification. *EDMS 772782 v.1*, 2006.

[3] H. Burkhardt and R Schmidt. Intensity and Luminosity after Beam Scraping. *CERN-AB-2004-032*, 2004.

[4] H. Burkhardt et al. Beam scraping for LHC injection, `http://documents.cern.ch/cgi-bin/setlink?base=preprint&categ=cern&id=lhcproj-2007-1022`. 2006.

[5] M Facchini, C Fischer, J.J Gras, S Hutchins, and R Jung. Scraping for LHC and collimation tests in the CERN SPS. Technical report, CERN, 2005.

[6] J. Lewis and W. Loftus. *Java Software Solutions*. Addison-Wesley, 4 edition, 2004.

[7] S. B. Pedersen. Unpublished private communication.

Appendix A

Dictionary

This appendix contains explanation of abbreviations and physics jargon used in this report.

A.1 Abbreviations

Administrative, etc

AB Accelerators and Beams department (at CERN)

ABP Accelerator Beam Physics group, under the AB department. This work was done in the ABP group.

CERN Conseil Europeen pour la Recherche Nucleaire (English: The European Organization for Nuclear Research)

CCC CERN Control Center

Accelerators and colliders

LEP Large Electron-Positron (collider), 1989–2000

LHC Large Hadron Collider

LINAC LINear ACcelerator

PS Proton Synchrotron

PSB Proton Synchrotron Booster

SPS Super Proton Synchrotron

SPS instruments

BCT Beam Current Transformer

BLM Beam Loss Monitor

Computers and programming

FEC Front End Computer

FESA Front End Software Architecture

GUI Graphical User Interface

JVM Java Virtual Machine – a virtual operating system, allowing byte-code compiled Java applications to run on any computer with a JVM implementation

OS Operating System

UML Unified Modeling Language, a general-purpose modeling language that specifies rules for drawing diagrams related to object oriented software development. The UML is specified by the Object Management Group

VME Versa Module Eurocard

Accelerator physics

MAD-X Methodical Accelerator Design, version 10. Software written and used at CERN. See also `http://mad.web.cern.ch/mad/`

MD Machine Development, experiment performed to learn more about the accelerator.

RMS Root Mean Square

A.2 Accelerator physics and beam optics dictionary

Bunch The protons (or other particles) inside one RF bucket.

Jaw (scraper) The **jaw** of a collimator or a scraper is the physical object which touches the beam to remove particles from it

Quench A magnet **quench** occurs whenever a magnet is heated sufficiently to make the superconducting coils lose their superconductivity. This leads to electrical resistance, which in turn leads to more heating, and possibly an avalanche effect.

RF Radio Frequency, used for electromagnetic waves with frequencies of about 3 Hz–30 GHz.

RF Bucket The (stable) area in phase space where particles are kept in the beam (longitudinal beam dynamics, RF acceleration).

Super cycle A **super cycle** consists of a repeating pattern of one or more **user cycles**.

User cycle A **user cycle** is the delivery of a beam with a given set of parameters to a target or another accelerator. For instance, the SPS can have a super cycle consisting of two user cycles for LHC injection, one user cycle for neutrino experiments, and one user cycle for another fixed target experiment.