15 April 2024

ITK online software Stage-2 proposal

ATLAS ITK

Summary

This document is a proposal for the modular design of the ITk online software Stage-2. The design fulfills the requirements of the integration, system test and commissioning groups. Notably, the proposed architecture is scalable to full detector size, and compatible with ATLAS TDAQ. The proposal is organized as a collaborative effort across the ITk community that encourages participation from new contributors. The planning and major milestones are aligned with the ITk schedule.

ATLAS Doc:	XXXX	
EDMS Id:	XXXX	
Version:	0.1	
Last modified:	`11:19 on 15 April 2024`	
Prepared by:	**Checked by:**	**Approved by:**
E. Antipov G. Brandt P. Morettini I. Siral C. Solans V. Tsiskaridze M. Wittgen		

Contents

List of Figures

List of Tables

1 Introduction

This document presents a comprehensive proposal for the design of the ITk online software Stage-2. The proposal is structured into several key sections, each addressing specific aspects of the software development process. Section 2 defines the design requirements. In Section 3, conceptual and technical guidelines are outlined, providing a framework for the development and implementation phases. The proposed distribution strategy is detailed in Section 4. Section 5 describes the review process. An overview of the software packages that constitute the design is presented in Section 6. The interfaces and interactions between the packages are outlined in Section 7. Section 9.1 specifies the deliverables of the project. Finally, the project schedule is outlined in Section 9, providing a timeline for key milestones and completion targets. Together, these sections provide a complete framework for developing and implementing the ITk online software Stage-2, ensuring that all requirements and project objectives are met.

2 Requirements

The proposal fulfills ten predefined requirements that ensure the compatibility of the architecture with other detector infrastructure, and specify both functional and non-functional aspects essential for the software's functionality. Requirements 8 - 10 expand the core requirements to extend the functionality and ease maintenance.

Requirement 1

Compatibility with the ATLAS Online Software infrastructure and release schedule

The ITk Stage-2 online software shall utilize the Control and Configuration tools [1] for control, communication, configuration and monitoring of the detector. If these tools do not fully meet operational requirements, consultation with the TDAQ online software group will be sought to address any issues collaboratively. All applications operating at Point 1 for detector operation or data acquisition must be based on the TDAQ release infrastructure [2] and build system [3]. Similarly, tools used during integration and commissioning phases shall adhere to these recommendations. Furthermore, the software shall be re-compiled against any new TDAQ release.

Requirement 2

Use tools and interfaces provided by FELIX, Data Handler and Central DCS

The ITk Stage-2 online software shall utilize the interfaces known at this point. FELIX-client for FELIX [4], SWROD [5] for the Data Handler, and Quasar [6] for Central DCS. In particular the production FELIX should operate as an always-on, stateless entity where the cards are initialized and configured immediately after power-up, followed by starting readout applications. Notably, DCS (Detector Control System) should not have the capability to reinitialize or restart FELIX. Orchestration tools shall be utilized

during integration and commissioning in collaboration with the DAQ (Data Acquisition) operations team.

Requirement 3

Develop tools for configuration and startup of lpGBT links

The ITk Stage-2 online software shall be responsible for the configuration and startup of lpGBT links following the FELIX startup. If any detector-specific startup actions need to be interleaved with portions of the FELIX startup sequence, consultation with the FELIX team will be sought to collaborate on finding a common solution.

Requirement 4

Provide specific quantities to be monitored and controlled via FELIX

The ITk Stage-2 online software shall monitor the health of the front-end electronics through the IC and EC e-links of the lpGBT links, as well as the transmission power of the optical links, the polarity, alignment and decoding status of e-links, the FELIX firmware version, and the status of the FELIX readout applications through the FELIX interface.

Requirement 5

Manage concurrent communication with the front-end for DAQ and DCS communication

The ITk Stage-2 online software shall manage any concurrent communication with the front-end for DAQ and DCS purposes. With the exception of trickle configuration (see below) FELIX shall not provide any arbitration mechanism for such access.

Requirement 6

Control the trickle configuration of the front-end via a FELIX API

The ITk Stage-2 online software shall manage (start, stop, update) the trickle configuration through the FELIX interface. The arbitration of the trickle data stream with respect to other streams shall be handled by the FELIX firmware.

Requirement 7

Provide a specific Data Handler plugin for processing and monitoring

The ITk Stage-2 online software shall provide a Data Handler plugin for aggregating input data into ROB fragments during physics runs. This plugin shall be responsible performing any detector-specific processing or monitoring activity required to support data taking before any completed ROB fragments are sent to the dataflow system. A fraction of events can potentially be decoded and made available for sampling, depending on the resultant processing load.

Requirement 8

Support detector integration, commissioning, and calibration.

The ITk Stage-2 online software shall provide calibration scans compatible with Loaded Local Supports and Integration activities, and will serve during the commissioning phase of the detector. Once the detector is in operation, these tools will be used for the calibration of the detector. The frequency of calibration and the duration of the calibration scans will be compatible with the inter-fill periods of the LHC. Results of the calibration will be stored in the conditions database for Run 4 [7]. Additional tools, configurations, and algorithms will be developed that will allow exercise of the dataflow model, multi-module running, parallel scanning, and multiple configurations running simultaneously.

Requirement 9

Provide tools to monitor the evolution of detector performance.

The ITk Stage-2 online software shall store the calibration results in the appropriate databases (configuration, calibration, conditions) with appropriate data formats. Additional tools will support the monitoring of the detector performance online, including those necessary for online data quality [8]. These tools will allow to carry out detector performance measurements throughout the life-cycle of the detector.

Requirement 10

Support Software Flexibility and Adaptability.

The ITk Stage-2 online software shall provide the ability to incorporate changes as needed throughout the development and operational phases. This includes accommodating updates to software configurations, algorithms, or interfaces based on evolving project requirements or feedback from operational use. Examples of development models are available from [9].

3 Guidelines

The guidelines for developing the ITk Stage-2 online software encompass both conceptual principles, focusing on developer-friendly design and software stability, and technical specifications particular the the management of individual packages.

3.1 Conceptual Guidelines

To ensure an easily maintainable and expandable code base that welcomes contributions from developers with diverse skill sets, the code design should prioritize developer-friendly practices, emphasizing software stability, performance, and reliability while maintaining flexibility to accommodate future changes. Coordination and management will be facilitated by organizing the ITk online software into distinct, well-defined, and independent packages, each open to contributions from individuals and institutions. Each package

within the ITk online software will have a dedicated coordinator responsible for its management and coordination.

The Unix philosophy of composability and modularity [10] is adopted as one of the guidelines. Tools will be developed with single functionality in mind, following the principle of "do one thing and one thing well". Groups of tools will be used together to build functionalities required for the different deliverables.

3.2 Technical Guidelines

Software packages will be publicly hosted repositories on CERN GitLab. Each repository will adhere to a standard structure, including a README file, a change log, and authors file, and comprehensive inline code documentation using Doxygen for C++ and python documentation standards. The structure of a standard package is show in in listing 1. In general, the TDAQ package layout conventions is preferred Individual packages can use other layouts if required. Unit tests should be provided with the aim to cover all code. Packages should be buildable with the TDAQ cmake-based build system but can also support standalone builds.

```
Tools                # Name of the package
  Tools              # Header files
    ToolA.hpp        # Class header
    ToolB.hpp        # Class header
  src                # Source files
    ToolA.cpp        # Class implementation
    ToolB.cpp        # Class implementation
    pyTools.cpp      # Python bindings
    test_ToolA.cpp   # Class test
    test_ToolB.cpp   # Class test
  python             # Python modules
  share              # Scripting tools
  data               # Additional files
  CMakeLists.txt     # CMake config file
  README
  CHANGELOG
  AUTHORS
```

Listing 1: Example package structure

Each package will feature continuous integration and automated build of artifacts, containers, and packages, alongside an internal release system with tagged stable versions. Interactions with other packages will utilize the API, as exposed on public header files (C++), or public methods (Python). Modification of the API will be responsibility of the distribution coordinators. Human readable documentation of the API will be required, see Section 4.4.

Additional technical guidelines are the SOLID principles for object oriented design [11]. Every class should have only one responsibility. Classes should be open for extension, but closed for modification. Functions that use pointers or references to base classes must be able to use objects of derived classes without knowing it. Clients should not be forced to

depend upon interfaces that they do not use. Dependence should be upon abstractions, not concretes.

4 Online software distribution

The ITk Stage-2 online software will be released in the form of a distribution of many packages which can be defined as an archive of the snapshots of the repositories accepted into the distribution. Technically it can be a list of tags and a collection of the binary release tarballs, including a collection of compose files. To help development of individual components, tools for dependency management (automated installation of binary dependencies) and setup of a suitable development environment should be supported. It should not be required to recompile dependencies that were not changed during development.

4.1 Distribution coordinators

The role of the distribution coordinators will be to enforce the development guidelines on the packages, add the packages to the distribution, and collect tags and release the distribution.

4.2 Package coordinators

The role of the package coordinators will be to follow the development guidelines, provide the necessary tests and continuous integration tools, present the packages for approval, and upgrade the packages after after any changes of the interfaces.

4.3 Certification

Certification will be performed on library/binary level by testing tools. Any package that passes certification should be included into the distribution. This means that certified packages with very similar functionality could be present in the distribution. For example, two versions of the same algorithm, one being faster and the other one more precise. The goal is to foster creativity, individual contributions, and complementary techniques.

4.4 Documentation

The API reference documentation will be collected automatically from the release from the comments in the source files, provided in a human readable way, and hosted on a public website [12]. For example a dedicated Doxygen interface for source code documentation in C++/Python, or using tools like MKDocs or Sphinx, to generate documentation from language agnostic API descriptions.

4.5 Release frequency

The distribution shall aim for regular release of development releases (once a week), as well as for major tagged releases aligned with milestones and functionality improvements.

In particular, the distribution shall be recompiled with every new TDAQ release (twice a year).

5 Review process

All contributions that fulfill the guidelines described in Section 3 will be welcome to be part of the ITk Stage-2 online software distribution. Proposals for new developments will be evaluated in terms of an importance-urgency matrix, see Tables 1, 2 and 3. Effort will be favoured for Required developments for the Primary target, while proposals for Optional developments for Extended targets will be based on a best effort basis.

	Primary	Main	Extended
Required			
Recommended			
Optional			

Table 1: Importance-Urgency matrix

Target	**Definition**
Primary	System tests, Loaded local supports
Main	Large scale tests, Point1
Extended	Comparisons, alternatives

Table 2: Importance scale of the importance-urgency matrix.

Urgency	**Definition**
Required	Required urgently for minimal functioning, right now.
Recommended	Required for smooth operation once minimal functionality is established.
Optional	Can be supported on a best effort basis in the long term.

Table 3: Urgency scale of the importance-urgency matrix.

Proposals for changes in the API an the definition of the packages will be accepted. Proposals can be made even to the review process itself. Proposals will transition from the proposed phase to a discussion phase before being accepted, see table 4. Throughout this process a continuous consultation with TDAQ and the ITk management is foreseen to help identify the placement in the importance-urgency matrix, and the acceptance of the development.

The discussion phase shall be public and open to all the ITk community. Proposals that follow the guidelines and can be clearly placed in the importance-urgency matrix will be rapidly accepted. Consultation with TDAQ and ITk management will be sought for all other cases.

Stage	Definition
Proposed	Initial layout. Not yet placed on the importance-urgency matrix.
Discussion	Iterating on proposal. Placed in the importance-urgency matrix.
Accepted	Approved. To be accepted for development.

Table 4: Review stages

6 Package Clusters

In order to identify the minimal set of software blocks for a working ITk online software Stage-2 project the following software package clusters are proposed.

- Register Map Package (Section 6.1)
- Hardware Package Cluster (Section 6.2)
- Front-End Data Encoder and Decoder Package Cluster (Section 6.3)
- Scan and Analysis Package Cluster (Section 6.4)
- FELIX Communication and Control Package Cluster (Section 6.5)
- lpGBT Communication Package (Section 6.6)
- OptoBoard and End of Stave Control Package Cluster (Section 6.7)
- FELIX and FE Emulator Package Cluster (Section 6.8)
- SWROD Plugins (Section 6.9)
- Trickle Configuration Package (Section 6.10)
- Front-end controllable Package Cluster (Section 6.11)
- DCS Package Cluster (Section 6.12)
- Database Interface Package Cluster (Section 6.13)
- Service Package Cluster (Section 6.14)

6.1 Register Map Package

This package will provide the definition of a hardware register, and a tool to generate them from a user readable (YAML) description. All ITk specific front-end register classes will be generated from the YAML description. This will provide a unified description and allow for easy validation of the registers across the project.

6.2 Hardware Package Cluster

One package will be provided for each front-end hardware in the ITk read-out chain. Common routines and interfaces will be used for all of them, including the register description, that will allow easy access to the configuration. They are expected to be a complete software representations of the underlying hardware.

6.3 Front-End Data Encoder and Decoder Package Clusters

The decoder is a stand-alone tool that converts the front-end raw data into hit-data. Similarly the encoder tool converts hit-data into front-end raw data. In both cases the front-end data is represented by a byte array, and the hit-data is a front-end specific data-structure. Thus at least one package per front-end type is foreseen containing both decoder and encoder tools. It is preferred that these packages have the minimal number of dependencies as possible to be able to evaluate their performance elsewhere or adapt them to other parts of the read-out chain. It is desired that the decoding process is as fast as the input data speed.

6.4 Scan and Analysis Package Clusters

Calibration of the front-end requires predefined data acquisition procedures (scan engines), and reconstruction algorithms (analysis) to characterize the front-end. These packages will provide the necessary scan engines and analysis classes for all the calibration scans. These packages will make use of the existing code base that will be adapted to the new interfaces.

6.5 FELIX Communication and Control Package Clusters

FELIX communication is handled by the FELIX client package, but for our software's to communicate properly we need software packages wrapping the FELIX client. This package will handle the transfer of data-packets coming from FELIX feeding them into relevant buffers and software packages for their decoding and processing. This package will be oblivious to the content of the data-received and will act mainly as broker. In addition this package will also be responsible for sending the commands and triggers to the FELIX system.

6.6 lpGBT Communication Package

LpGBT communication between FELIX is handled by the IC communication protocol. This IC protocol can also be used for lpGBT to transfer I2C commands to other electronics inside our detectors. Since pixels, strips and even other ATLAS detectors like BCM Prime utilizes this lpGBT structure, we propose an lpGBT communication package that will also handle the I2C, EC communications of different front-ends. This package will be designed as an arbitrator between other hardware packages that will be using the I2C and EC protocols, and will be providing the future DCS packages a basis for communication.

6.7 OptoBoard and Strips EoS Control Package

This package will host the configuration routines for the LpGBT and other front-ends in the ITk pixel and strips optical readout system. It should encompass the the LpGBT communication and hardware packages as described in Section **??**. Existing OptoBoard software will fall into this category.

6.8 FELIX and FE Emulator Package Clusters

This packages should emulate a FELIX server, and mimic different ITk front-end by sending appropriate responses to the commands send to them. The goal of these packages is to create a software environment where users and groups can test different software packages.

6.9 SWROD Plugins

This package will provide a software plugin interfacing the other packages listed above with the ATLAS TDAQ system.

6.10 Trickle Configuration Package

This package provides a common trickle configuration routines. The structure of these packages will be clarified in the future.

6.11 Front-end controllable Package Clusters

This is meant for data taking. Configure the front-end before the start of the run. Receive status messages from monitoring applications about the modules. Carry out stopples recovery actions, and module reconfiguration. Might require interaction with DCS.

6.12 DCS Package Cluster

ITk DCS system will be required to be interfaced with FELIX so that the detector status can be directly monitored by the DCS system. This package will control and regulate the DCS communication with the ITk system. For Strips architecture, this system is also expected to provide an interlock system [13] by enabling and disabling different staves.

6.13 Database Interface Package Cluster

The ITk project has multiple databases such as configuration, results and conditions database. The ITk online software will need to communicate with these databases to setup the scans and/or to store the results of the scans. Database interface packages will establish the communications between these packages.

6.14 Service Package Cluster

Many small high-level service packages will be designed to control the ITk online software as well as software for making it easy for users to interact. All of these extra software units are grouped under the service packages umbrella.

7 Interfaces

As described above, the proposed ITk software is composed of packages and these packages would need to be interfaced together with common and predefined structures. In Figure 1, the overall package connection skeleton is shown. Each of the shown connections between the packages have a predefined interfaces. The definitions of these interference's will be defined by the community. Preliminary proposals for some the interfaces are summarised below.

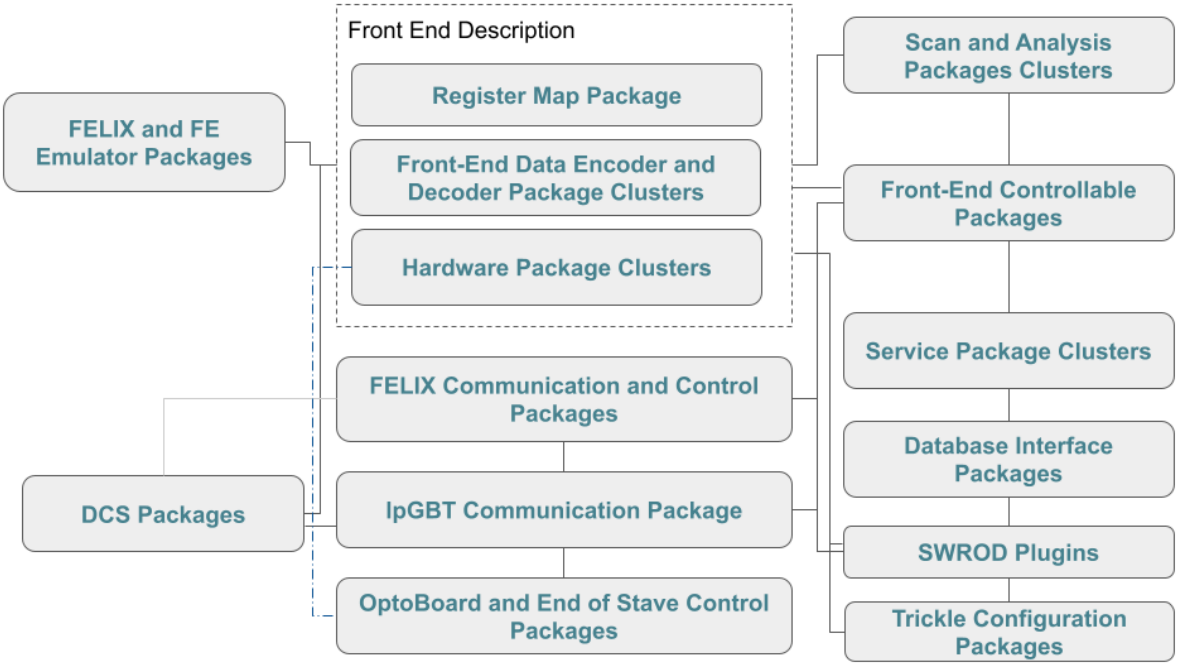

Figure 1: ITk software overall structure with interface links

7.1 Hardware Interfaces

The hardware interfaces library will be the base library that will be used for generating the software representations ITk hardware/FE. Each HW will contain register structures (assuming it has one), structures to generate bits streams for commands, structure that will decode/encode the data as described in Section 7.2. The main requirements for these structures are:

- Layered Structure

- Single class for single feature approach.

7.2 Data Interface for Front-End Data Encoder and Decoder Packages

The Encoder and decoder package sits at the hearth of the ITk software project, and it's designed to decode raw data that is coming from the FELIX client package and output the decoded hit data into the decoded hit data structure that is described below.

In parallel, the same tool is expected to input this decoded data structure and convert it into the raw data.

7.2.1 Decoded Hit Data Structure

For decoded hit information, the proposed data-interface, ITk software project proposes to use the ATLAS raw event data format [14]. This structure is composed of a fixed header that contains crucial data for processing the enclosed data, a body that can be defined as big as necessary and where the size is predefined in the header and a trailer for handling check sums. One such data block is called a fragment, and the structure allows to host different fragment blocks inside the body of a fragment. In other words, these fragments can be nested together.

The proposal is to have a nested multiple fragments for all decoded data-packets, where one fragment contains the necessary FE configuration, one fragment contains the decoded hits and other fragments can be introduced for all other data formats.

A C++ structure will built around this data-structure for easy processing and integration, so majority of the users won't need to deal with the data-format directly.

7.3 Data Interface for FELIX Communication and Control Packages

The FELIX client which is at the heart of FELIX communication and control package provides raw-data. FELIX communication package will act as a broker and will provide the raw data to the other software packages. It's expected that a small set of extra data packages will be added to he raw data for passing extra information like FID of the data packet.

The more complicated part is the command stream due to limitations on the command transfer speeds, timings and requirements for orchestration. For this reason a command class is proposed. Where sent commands in raw format will be embedded inside this class. This command class will be able to handle how this data should be transferred. The user will provide a vector of command class pointers to the software when they want to transmit data via FELIX.

In addition to the input and output data, the FELIX communication is also required to open/configure FELIX communication channels. These communication packages will be opened enabled by calling the relevant class functions.

8 Management structure

The general management structure proposal can be view in the figure 2. At the top there is a top level management group whose job is to define the interfaces and control structure. Handle the software publication strategy and coordinate the sub-groups. The top level management group job will not be micromanaging each package, but will be there to steer the big project with the big picture.

After the top management, there will be a flat distribution where each group will be responsible of different software packages. Each group is expected to have a coordinator, who among other tasks will be responsible for the integration with other software packages. As an example the histogrammer will have an explicit interface to the scan controller and the coordinators of both groups will be required to communicate with each other as well as with the top level management to define these interfaces.

As the man power is limited, the proposal assumes that each group will take charge of multiple packages. In the future, if there is man power and structural needs, these groups can be reduced to groups of smaller sets.

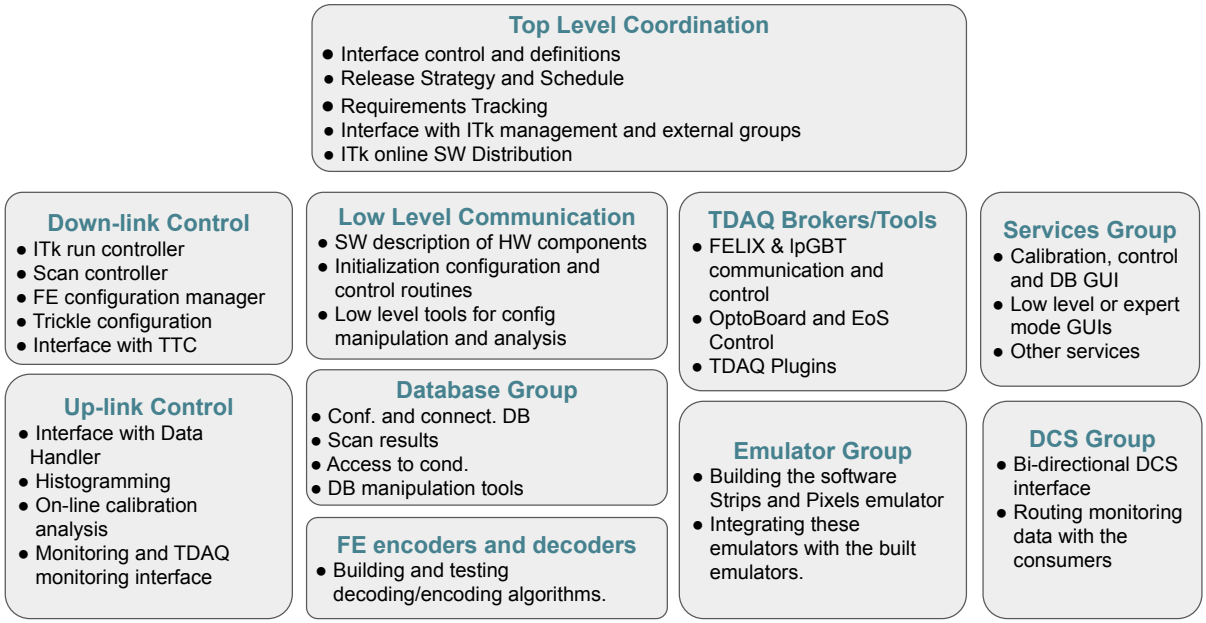

Figure 2: Proposed management structure

9 Schedule

This Section describes the initial high-level schedule for the ITk Stage-2 online software. A bottom-up approach is used to build the schedule, taking into account deliverables listed in Section 9.1, and the inter-dependencies with other parts of the project, such as the integration date of the first barrel layers in SR1 in 2024, the release data of the API

for the Data Handler in 2027, and the availability of the TDAQ release for the start of Run 4 at the beginning of 2028.

A summary of the schedule for ITk Stage-2 online software project can be found in Figure 3, and divided into different Work Breakdown Structures (WBS) detailed in 9.2. A large amount of float is accumulated in Release 6 in 2026. This aims to capture project delays before the PRR in Q4 2026, and before the TDAQ build candidate for Run 4 in Q1 2027. The effort required is over 4 FTEs, with more effort at the beginning of the project, less towards 2026, and a slight increase in 2028, as the final configuration cannot be implemented until the detector is in Point 1. This means that there is a large amount of float in 2026.

The schedule includes dates for project reviews outlined in Section 9.3. This plans will evolve as the project moves forward, including the addition of detailed breakdown of tasks. This will allow the ITk management to track progress, identify delays as early as possible and take preventive actions.

Task Name	Duration	Start	Finish
Proposal	**6.2 wks**	**Fri 01/03/24**	**Mon 15/04/24**
Release 0	10.4 wks	Mon 25/03/24	Tue 04/06/24
Release 1	**9 wks**	**Fri 03/05/24**	**Fri 05/07/24**
Release 2	**14 wks**	**Fri 14/06/24**	**Fri 20/09/24**
Release 3	**24 wks**	**Fri 23/08/24**	**Fri 07/02/25**
Release 4	**16 wks**	**Sat 01/02/25**	**Fri 23/05/25**
Release 5	**24.2 wks**	**Mon 26/05/25**	**Mon 10/11/25**
Release 6	**29 wks**	**Sun 01/02/26**	**Fri 21/08/26**
Release 7	**9 wks**	**Mon 01/02/27**	**Fri 02/04/27**
Release 8	**30 wks**	**Tue 01/02/28**	**Mon 28/08/28**

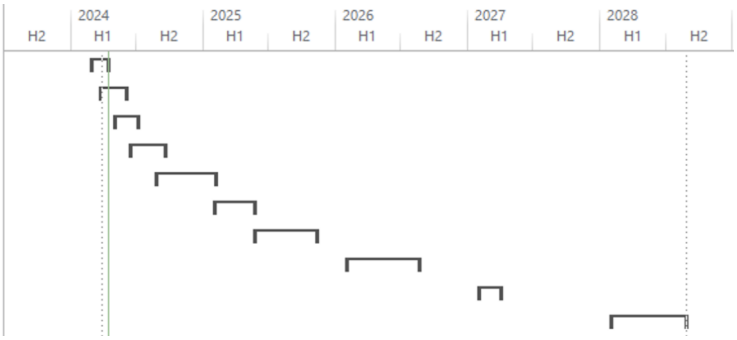

Figure 3: Project schedule

9.1 Deliverables

The deliverables of the ITk Stage-2 online software in order of delivery are the following:

- Software distribution
- Interfaces between different blocks
- Certification guidelines and tests
- FELIX arbitration mechanism
- Calibration scans
- Calibration scan supervision
- Front-end configuration interface
- Interface with front-end configuration and storage of results
- Preliminary detector description for SR1

- Monitoring algorithms
- SWROD plugins for data processing
- SWROD plugins for histogramming
- DQ histogramming algorithms
- ALTI configuration
- Trickle configuration mechanism
- Conditions database integration
- Detector description for SR1
- Detector description for Point 1

9.2 Work Breakdown Structures

The current schedule foresees 10 Work Breakdown Structures (WBS) for the project. Each one providing different deliverables. The duration of each WBS are between 9 and 30 weeks.

Release 0 will deliver the first software distribution, built after the initial interface definitions are approved, and the certification guidelines and tests for the packages are in place. It is expected to test the software in a single module setup for Strips and Pixels (Section 10.1). The project will be ready for a Specifications Review by Q3 2024.

Release 1 will deliver the first set of calibration scans for integration. Using the interfaces from Release 0, it will contain the first version of the scan definitions needed by Pixels and Strips. Notably it will provide first version of the decoder/encoder packages. It will be tested in a vertical slice setup (Section 10.2).

Release 2 will deliver the first version of the FELIX broker packages (DCS arbitration), the next version of the Pixel and Strips scans for large structures, and the preliminary configuration database definition of the large structures in SR1. It will be tested in a multi-module setups setup (Section 10.1).

Release 3 will be built for the Strip Stave tests expected in SR1 by Q3 2024. A large amount of effort is expected to adapt the tools to the real hardware. It will be exercised in all three types of testing setups as described in Section **??**. It will be also the basis used for a Preliminary Design Review in Q1 2025.

Release 4 will target Pixel loaded local supports in SR1. It will deliver TDAQ integration of the detector with the first SWROD plugins and histogramming and monitoring tools. It will be delivered by Q3 2025, and will be mainly exercised on multiple Vertical Slice tests setups (Section 10.2).

Release 5 will target Triggering configuration using ALTI protypes [15], and detector description in SR1 to operate multiple FELIX simultaneously, as well as DQ tools. A large period of testing is expected for this release. It will be exercised using high-performance setups (Section 10.3), and ready for a Final Design Review in Q4 2025.

Release 6 will be built in Q1 2026, and will deliver the GUI tools for calibration and configuration for data-taking using TDAQ tools. It will also deliver the first version of the Trickle configuration tools. It will be tested in Vertical Slice setups (Section 10.2). This will be the basis for the Production Readiness Review in Q4 2026.

Release 7 will be built in Q1 2027 and will deliver the conditions database interfaces, as well as the full description of the ITk in SR1. It is expected that the interfaces with TDAQ and and the conditions database are well established at this point. This will be the first version of the ITk Stage-2 software ready to deploy in Point 1.

Release 8 will be built in Q1 2028. It will deliver the description of the ITk detector for operations in the TDAQ configuration database, including the relationships to the TDAQ infrastructure equipment. It is expected to have a month of testing period of the release in Point 1 before it is ready for the commissioning of the detector, and a month of testing/commissioning of the detector once the first Strip barrel is lowered into UX15. At which point the project will be delivered to the operations team of the ITk. No changes to the interfaces are expected at this point.

9.3 Project reviews

The ITk Stage-2 online software project will have four reviews like the rest of the ITk, a Specifications Review (SPR) in Q3 2024, a Preliminary Design Review (PDR) in Q1 2025, a Final Design Review (FDR) in Q4 2025, and a Production Readiness Review (PRR) in Q4 2026. These reviews will be internal to ITk formed by members of the management and experts from TDAQ Controls and Readout, ATLAS DCS, and the current Inner Detector.

9.3.1 Specifications Review

A set of specifications for the proposed ITk Stage-2 online software will be prepared as a document based on the requirements outlined in this document, and additional feedback from Loaded Local Supports and Integration experts. The SPR will evaluate the that the document describes the required functionality and interfaces.

9.3.2 Preliminary Design Review

The PDR will establish if the design meets all the aspects of the specifications. It will evaluate the prototypes for the calibration scans, the DCS arbitration, and the decoder packages based on results from Vertical Slice test setups (see Section 10.2). Other components like the SWROD plugins, histogramming, DQ, ALTI, and databases will be made based on preliminary implementations using the current interfaces. This will allow to identify if the interfaces are suitable or need modification.

9.3.3 Final Design Review

The FDR will assess whether the design is mature enough for Integration. All requirements and specifications will be reviewed, including performance measurements carried

out in SR1 of all the components, including the SWROD plugins, the histogramming, and DQ tools, and ALTI integration.

9.3.4 Production Readiness Review

The PRR will assess the readiness of the components for the operation of ITk based on the characterization results of the detector during Integration. This includes the detailed procedure followed during the calibration of parts of the detector, and the performance measurements measured in terms of resource utilization (CPU load, timing, network bandwidth) of FELIX and the DataHandler. This will provide valuable information to the operations team in view of the long term operation of the detector.

10 Testing infrastructure

This section outlines the testing infrastructure expected for the development of the ITk Stage-2 online software. These are increasingly complex systems that allow testing from a single module to a fully loaded FELIX server. The FELIX hardware discussed in this section corresponds to the FELIX Phase-I card (FLX-712), the FELIX Phase-I server (SuperMicro SYS-5028R-WR), and the SWROD Phase-I server (SuperMicro SYS-1029P-WTRT) that are already available in ITk.

10.1 Single Module Setup

A single module setup is the minimal read-out chain from the front-end electronics to FELIX, and thus to the ITk Stage-2 online software. This setup serves the purpose for testing of low level communication tools, calibration scan procedures, monitoring tools, SWROD plugin implementations, histogramming and DQ algorithms, graphical interfaces. The components of the setup are the following:

- One or many Front-end modules, up to one Loaded Local Support.
- Laboratory Power supplies with minimal supervision and safety interlocks.
- Cooling infrastructure compatible with the number of modules.
- Optical to electrical converter such as VLDB+, OptoBoard or EoS.
- Break out boards and or interference cards for the appropriate module.
- Fiber optic cables (MPO-24/12 break out cables) for the appropriate optical adapter.
- FELIX card (FLX-712) and a FELIX Phase-I server.

10.2 Vertical slice

A vertical slice setup is the minimal production ready read-out chain from the front-end electronics to FELIX. This setup serves the purpose of characterizing the detector in realistic conditions, and thus evaluating the software under those same conditions. These include the testing of the complete readout chain. The components of the setup are the following:

- One Loaded Local Support.
- Production power supplies and control systems
- Production services.
- Production OptoBoard or EoS.
- Fiber optic cables (MPO-24/12 break out cables) for the appropriate optical adapter.
- FELIX card (FLX-712) and a FELIX Phase-I server
- 100 Gbps (Pixel) or 25 Gbps (Strips) network interface card
- 100 Gbps network switch
- SWROD Phase-I server

10.3 High Performance Setup

A High Performance Setup is a Vertical Slice with many links connected to the same FELIX. This setup allows to test the read-out chain in full load conditions. The components of the setup are the following:

- Many Loaded Local Support.
- Production power supplies and control systems
- Production services.
- Production OptoBoard or EoS.
- Fiber optic cables (MPO-24/12 break out cables) for the appropriate optical adapter.
- FELIX card (FLX-712) and a FELIX Phase-I server
- 100 Gbps (Pixel) or 25 Gbps (Strips) network interface card
- 100 Gbps network switch
- SWROD Phase-I server

10.4 High Performance Emulator Setup

This setup serves the same purpose of a High Performance Setup with Emulators.

References

[1] A. Kazarov *et al.*, "The Controls and Configuration Software of the ATLAS Data Acquisition System: evolution towards LHC Run 3," in *EPJ Web Conf. 251*, 2021. 04019.

[2] "ATLAS Trigger/DAQ Software." Online https://atlas-tdaq-sw-releases.web.cern.ch. Accessed 2024-04-12.

[3] "CMake for ATLAS TDAQ Software." Twiki https://twiki.cern.ch/twiki/bin/viewauth/Atlas/DaqHltCMake. Accessed 2024-04-12.

[4] "ATLAS FELIX Client." Online https://gitlab.cern.ch/atlas-tdaq-felix/felix-client. Accessed 2024-04-12.

[5] "ATLAS SWROD." Online https://gitlab.cern.ch/atlas-tdaq-software/swrod. Accessed 2024-04-12.

[6] P. Nikiel *et al.*, "quasar : The Full-Stack Solution for Creation of OPC-UA Middleware," in *ICALEPCS*, 2019. MOPHA100.

[7] M. Mineev *et al.*, "Evolution of the ATLAS CREST conditions DB project," tech. rep., CERN, Geneva, 2023.

[8] "ATLAS Data Quality Monitoring Display." Online https://gitlab.cern.ch/atlas-tdaq-software/dqm_display. Accessed 2024-04-12.

[9] " CERN's Open Source Program Office ." Online https://opensource.web.cern.ch/. Accessed 2024-04-12.

[10] "UNIX Phylosophy." Online https://en.wikipedia.org/wiki/Unix_philosophy. Accessed 2024-04-12.

[11] "SOLID Principles." Online https://en.wikipedia.org/wiki/SOLID. Accessed 2024-04-12.

[12] "WebEOS hosting at CERN." Online https://webeos.docs.cern.ch. Accessed 2024-04-12.

[13] "ITk System Interlock Strategy," tech. rep., CERN, 2023. https://edms.cern.ch/document/2387552.

[14] "The raw event format in the ATLAS Trigger and DAQ." CERN EDMS https://edms.cern.ch/document/ATL-D-ES-0019. Accessed: 2020-04-06.

[15] "ATLAS Local Trigger Interface." Twiki https://twiki.cern.ch/twiki/bin/viewauth/Atlas/LevelOneCentralTriggerALTI. Accessed 2024-04-12.