**Sentinel-2 Agriculture**

Design Definition File



|  |  |
| --- | --- |
| Milestone | Milestone 2 |
| Version | 1.0 |
| Authors | CESBIO - Olivier Hagolle, Jordi Inglada, Silvia Valero, Marcela Arias, David Morin  CS - Mickael Savinaud, Alexia Mondot, Aurélien Bricier  UCL - Sophie Bontemps, Guadalupe Sepulcre Canto, Nicolas Matton  CS ROMANIA - Cosmin Cara, Cosmin Udroiu |

***This page is intentionally blank***

***Table of recorded changes***

|  |  |  |
| --- | --- | --- |
| Issue/Rev. | Date | Reason |
| 0.1 |  | Template |
| 0.2 |  | First preliminary version |
| 0.3 |  | Second preliminary version for consortium review |
| 1.0 | 21/04/2015 | First version for ESA review |

***Table of recorded changes***

**Issue Record Sheet**

|  |  |  |
| --- | --- | --- |
| **ISSUE** | **DATE** | **REASON** |
| 1.0 | 21 April 2015 | Initial version |

***Table of contents***

[1 Introduction 18](#_Toc417423419)

[1.1 Purpose and Scope 18](#_Toc417423420)

[1.2 Structure of the Document 18](#_Toc417423421)

[1.3 References 18](#_Toc417423422)

[1.3.1 Applicable Documents 18](#_Toc417423423)

[1.3.2 Reference Documents 19](#_Toc417423424)

[1.3.3 Acronyms and Abbreviations 19](#_Toc417423425)

[2 System Algorithm Theoretical Basis Documents 21](#_Toc417423426)

[3 Design overview 22](#_Toc417423427)

[3.1 System static architecture 22](#_Toc417423428)

[3.1.1 User Interface Layer 22](#_Toc417423429)

[3.1.1.1 Processing System Configuration Utility 23](#_Toc417423430)

[3.1.1.2 Execution Dashboard 24](#_Toc417423431)

[3.1.1.3 User Subscription Manager 26](#_Toc417423432)

[3.1.2 Service Layer 26](#_Toc417423433)

[3.1.2.1 L1C Product Downloader 26](#_Toc417423434)

[3.1.2.2 L1T Product Downloader 26](#_Toc417423435)

[3.1.2.3 The Orchestrator 26](#_Toc417423436)

[3.1.2.4 Scheduler 27](#_Toc417423437)

[3.1.2.5 Resource Manager 27](#_Toc417423438)

[3.1.2.6 Processors 27](#_Toc417423439)

[3.1.2.7 Modules 29](#_Toc417423440)

[3.1.2.8 Archiver 29](#_Toc417423441)

[3.1.2.9 Persistence Manager 29](#_Toc417423442)

[3.1.3 Persistence Layer 30](#_Toc417423443)

[3.1.3.1 Configuration Database 30](#_Toc417423444)

[3.1.3.2 Execution information database 30](#_Toc417423445)

[3.1.3.3 Processing Area (File System) 30](#_Toc417423446)

[3.1.3.4 Archive (File System) 30](#_Toc417423447)

[3.1.4 System static package and library architecture 30](#_Toc417423448)

[3.2 System Dynamic Architecture 31](#_Toc417423449)

[3.3 System Behaviour 33](#_Toc417423450)

[3.3.1 On-Demand Processing 33](#_Toc417423451)

[3.4 Interfaces Context 34](#_Toc417423452)

[3.4.1 ESA Sentinel-2 Interface 34](#_Toc417423453)

[3.4.2 USGS LandSat8 Interface 35](#_Toc417423454)

[3.4.3 User Interface 35](#_Toc417423455)

[3.4.4 Orchestrator Interface 36](#_Toc417423456)

[3.4.5 Sentinel-2 Toolbox Interface 36](#_Toc417423457)

[3.5 Specific constraint 37](#_Toc417423458)

[3.5.1 SLURM 37](#_Toc417423459)

[3.5.2 D-Bus 37](#_Toc417423460)

[3.5.3 Cron 37](#_Toc417423461)

[3.6 Memory and CPU Budget 37](#_Toc417423462)

[3.7 Design Standards, Conventions and Procedures 38](#_Toc417423463)

[4 System Design 39](#_Toc417423464)

[4.1 Overall Architecture 39](#_Toc417423465)

[4.2 Components Design 40](#_Toc417423466)

[4.2.1 Orchestrator Component 41](#_Toc417423467)

[4.2.1.1 Type 41](#_Toc417423468)

[4.2.1.2 Purpose 41](#_Toc417423469)

[4.2.1.3 Function 41](#_Toc417423470)

[4.2.1.4 Subordinates 42](#_Toc417423471)

[4.2.1.5 Dependencies 42](#_Toc417423472)

[4.2.1.6 Interfaces 42](#_Toc417423473)

[4.2.1.7 Resources 44](#_Toc417423474)

[4.2.1.8 References 44](#_Toc417423475)

[4.2.1.9 Data 44](#_Toc417423476)

[4.2.2 Scheduler Component 44](#_Toc417423477)

[4.2.2.1 Type 44](#_Toc417423478)

[4.2.2.2 Purpose 45](#_Toc417423479)

[4.2.2.3 Function 45](#_Toc417423480)

[4.2.2.4 Subordinates 45](#_Toc417423481)

[4.2.2.5 Dependencies 45](#_Toc417423482)

[4.2.2.6 Interfaces 45](#_Toc417423483)

[4.2.2.7 Resources 46](#_Toc417423484)

[4.2.2.8 References 46](#_Toc417423485)

[4.2.2.9 Data 46](#_Toc417423486)

[4.2.3 L1C Product Downloader 46](#_Toc417423487)

[4.2.3.1 Type 46](#_Toc417423488)

[4.2.3.2 Purpose 46](#_Toc417423489)

[4.2.3.3 Function 46](#_Toc417423490)

[4.2.3.4 Subordinates 47](#_Toc417423491)

[4.2.3.5 Dependencies 49](#_Toc417423492)

[4.2.3.6 Interfaces 49](#_Toc417423493)

[4.2.3.7 Resources 49](#_Toc417423494)

[4.2.3.8 References 49](#_Toc417423495)

[4.2.3.9 Data 50](#_Toc417423496)

[4.2.4 L1T Product Downloader 50](#_Toc417423497)

[4.2.4.1 Type 50](#_Toc417423498)

[4.2.4.2 Purpose 50](#_Toc417423499)

[4.2.4.3 Function 50](#_Toc417423500)

[4.2.4.4 Subordinates 50](#_Toc417423501)

[4.2.4.5 Dependencies 51](#_Toc417423502)

[4.2.4.6 Interfaces 51](#_Toc417423503)

[4.2.4.7 Resources 51](#_Toc417423504)

[4.2.4.8 References 52](#_Toc417423505)

[4.2.4.9 Data 52](#_Toc417423506)

[4.2.5 Compute Tile Lists From AOI Module 52](#_Toc417423507)

[4.2.5.1 Type 52](#_Toc417423508)

[4.2.5.2 Purpose 52](#_Toc417423509)

[4.2.5.3 Function 52](#_Toc417423510)

[4.2.5.4 Subordinates 52](#_Toc417423511)

[4.2.5.5 Dependencies 52](#_Toc417423512)

[4.2.5.6 Interfaces 52](#_Toc417423513)

[4.2.5.7 Resources 53](#_Toc417423514)

[4.2.5.8 References 53](#_Toc417423515)

[4.2.5.9 Data 53](#_Toc417423516)

[4.2.6 DEM&WB Processing Module 53](#_Toc417423517)

[4.2.6.1 Type 53](#_Toc417423518)

[4.2.6.2 Purpose 53](#_Toc417423519)

[4.2.6.3 Function 54](#_Toc417423520)

[4.2.6.4 Subordinates 54](#_Toc417423521)

[4.2.6.5 Dependencies 57](#_Toc417423522)

[4.2.6.6 Interfaces 57](#_Toc417423523)

[4.2.6.7 Resources 57](#_Toc417423524)

[4.2.6.8 Data 58](#_Toc417423525)

[4.2.7 L2A S2 Pre Processing Module 58](#_Toc417423526)

[4.2.7.1 Type 58](#_Toc417423527)

[4.2.7.2 Purpose 58](#_Toc417423528)

[4.2.7.3 Function 58](#_Toc417423529)

[4.2.7.4 Subordinates 58](#_Toc417423530)

[4.2.7.5 Dependencies 58](#_Toc417423531)

[4.2.7.6 Interfaces 58](#_Toc417423532)

[4.2.7.7 Resources 59](#_Toc417423533)

[4.2.7.8 References 59](#_Toc417423534)

[4.2.7.9 Data 59](#_Toc417423535)

[4.2.8 L2A MACCS Processing Module 60](#_Toc417423536)

[4.2.8.1 Type 60](#_Toc417423537)

[4.2.8.2 Purpose 60](#_Toc417423538)

[4.2.8.3 Function 60](#_Toc417423539)

[4.2.8.4 Subordinates 60](#_Toc417423540)

[4.2.8.5 Dependencies 60](#_Toc417423541)

[4.2.8.6 Interfaces 60](#_Toc417423542)

[4.2.8.7 Resources 61](#_Toc417423543)

[4.2.8.8 References 61](#_Toc417423544)

[4.2.8.9 Data 61](#_Toc417423545)

[4.2.9 L2A L8 Post Processing Module 61](#_Toc417423546)

[4.2.9.1 Type 61](#_Toc417423547)

[4.2.9.2 Purpose 61](#_Toc417423548)

[4.2.9.3 Function 61](#_Toc417423549)

[4.2.9.4 Subordinates 62](#_Toc417423550)

[4.2.9.5 Dependencies 62](#_Toc417423551)

[4.2.9.6 Interfaces 62](#_Toc417423552)

[4.2.9.7 Resources 63](#_Toc417423553)

[4.2.9.8 References 63](#_Toc417423554)

[4.2.9.9 Data 63](#_Toc417423555)

[4.2.10 Bidirectional Correction Processor 63](#_Toc417423556)

[4.2.10.1 Type 63](#_Toc417423557)

[4.2.10.2 Purpose 63](#_Toc417423558)

[4.2.10.3 Function 64](#_Toc417423559)

[4.2.10.4 Subordinates 64](#_Toc417423560)

[4.2.10.5 Dependencies 67](#_Toc417423561)

[4.2.10.6 Interfaces 67](#_Toc417423562)

[4.2.10.7 Resources 68](#_Toc417423563)

[4.2.10.8 References 68](#_Toc417423564)

[4.2.10.9 Data 68](#_Toc417423565)

[4.2.11 Iterative Composite Synthesis Processor 68](#_Toc417423566)

[4.2.11.1 Type 68](#_Toc417423567)

[4.2.11.2 Purpose 68](#_Toc417423568)

[4.2.11.3 Function 69](#_Toc417423569)

[4.2.11.4 Subordinates 69](#_Toc417423570)

[4.2.11.5 Dependencies 72](#_Toc417423571)

[4.2.11.6 Interfaces 72](#_Toc417423572)

[4.2.11.7 Resources 74](#_Toc417423573)

[4.2.11.8 References 74](#_Toc417423574)

[4.2.11.9 Data 74](#_Toc417423575)

[4.2.12 Compute LAI 74](#_Toc417423576)

[4.2.12.1 Type 74](#_Toc417423577)

[4.2.12.2 Purpose 74](#_Toc417423578)

[4.2.12.3 Function 75](#_Toc417423579)

[4.2.12.4 Subordinates 75](#_Toc417423580)

[4.2.12.5 Dependencies 76](#_Toc417423581)

[4.2.12.6 Interfaces 76](#_Toc417423582)

[4.2.12.7 Resources 77](#_Toc417423583)

[4.2.12.8 References 77](#_Toc417423584)

[4.2.12.9 Data 77](#_Toc417423585)

[4.2.13 Reprocess LAI 77](#_Toc417423586)

[4.2.13.1 Type 77](#_Toc417423587)

[4.2.13.2 Purpose 77](#_Toc417423588)

[4.2.13.3 Function 77](#_Toc417423589)

[4.2.13.4 Subordinates 78](#_Toc417423590)

[4.2.13.5 Dependencies 78](#_Toc417423591)

[4.2.13.6 Interfaces 78](#_Toc417423592)

[4.2.13.7 Resources 79](#_Toc417423593)

[4.2.13.8 References 79](#_Toc417423594)

[4.2.13.9 Data 79](#_Toc417423595)

[4.2.14 NDVI Metrics 79](#_Toc417423596)

[4.2.14.1 Type 79](#_Toc417423597)

[4.2.14.2 Purpose 79](#_Toc417423598)

[4.2.14.3 Function 79](#_Toc417423599)

[4.2.14.4 Subordinates 80](#_Toc417423600)

[4.2.14.5 Dependencies 80](#_Toc417423601)

[4.2.14.6 Interfaces 80](#_Toc417423602)

[4.2.14.7 Resources 81](#_Toc417423603)

[4.2.14.8 References 81](#_Toc417423604)

[4.2.14.9 Data 81](#_Toc417423605)

[4.2.15 Samples Preparation 81](#_Toc417423606)

[4.2.15.1 Type 81](#_Toc417423607)

[4.2.15.2 Purpose 82](#_Toc417423608)

[4.2.15.3 Function 82](#_Toc417423609)

[4.2.15.4 Subordinates 82](#_Toc417423610)

[4.2.15.5 Dependencies 84](#_Toc417423611)

[4.2.15.6 Interfaces 85](#_Toc417423612)

[4.2.15.7 Resources 85](#_Toc417423613)

[4.2.15.8 References 85](#_Toc417423614)

[4.2.15.9 Data 86](#_Toc417423615)

[4.2.16 Compute temporal features module 86](#_Toc417423616)

[4.2.16.1 Type 86](#_Toc417423617)

[4.2.16.2 Purpose 86](#_Toc417423618)

[4.2.16.3 Function 86](#_Toc417423619)

[4.2.16.4 Subordinates 86](#_Toc417423620)

[4.2.16.5 Dependencies 87](#_Toc417423621)

[4.2.16.6 Interfaces 88](#_Toc417423622)

[4.2.16.7 Resources 88](#_Toc417423623)

[4.2.16.8 References 88](#_Toc417423624)

[4.2.16.9 Data 88](#_Toc417423625)

[4.2.17 Learning 89](#_Toc417423626)

[4.2.17.1 Type 89](#_Toc417423627)

[4.2.17.2 Purpose 89](#_Toc417423628)

[4.2.17.3 Function 89](#_Toc417423629)

[4.2.17.4 Subordinates 89](#_Toc417423630)

[4.2.17.5 Dependencies 90](#_Toc417423631)

[4.2.17.6 Interfaces 90](#_Toc417423632)

[4.2.17.7 Resources 91](#_Toc417423633)

[4.2.17.8 References 91](#_Toc417423634)

[4.2.17.9 Data 91](#_Toc417423635)

[4.2.18 Classification and Validation Module 91](#_Toc417423636)

[4.2.18.1 Type 91](#_Toc417423637)

[4.2.18.2 Purpose 91](#_Toc417423638)

[4.2.18.3 Function 92](#_Toc417423639)

[4.2.18.4 Subordinates 92](#_Toc417423640)

[4.2.18.5 Dependencies 93](#_Toc417423641)

[4.2.18.6 Interfaces 93](#_Toc417423642)

[4.2.18.7 Resources 94](#_Toc417423643)

[4.2.18.8 References 94](#_Toc417423644)

[4.2.18.9 Data 94](#_Toc417423645)

[4.2.19 LUT Generation 94](#_Toc417423646)

[4.2.19.1 Type 94](#_Toc417423647)

[4.2.19.2 Purpose 94](#_Toc417423648)

[4.2.19.3 Function 94](#_Toc417423649)

[4.2.19.4 Subordinates 94](#_Toc417423650)

[4.2.19.5 Dependencies 95](#_Toc417423651)

[4.2.19.6 Interfaces 95](#_Toc417423652)

[4.2.19.7 Resources 96](#_Toc417423653)

[4.2.19.8 References 96](#_Toc417423654)

[4.2.19.9 Data 96](#_Toc417423655)

[4.2.20 Time Series Feature Extraction Module 96](#_Toc417423656)

[4.2.20.1 Type 96](#_Toc417423657)

[4.2.20.2 Purpose 96](#_Toc417423658)

[4.2.20.3 Function 96](#_Toc417423659)

[4.2.20.4 Subordinates 97](#_Toc417423660)

[4.2.20.5 Dependencies 98](#_Toc417423661)

[4.2.20.6 Interfaces 99](#_Toc417423662)

[4.2.20.7 References 99](#_Toc417423663)

[4.2.20.8 Data 99](#_Toc417423664)

[4.2.21 Resampling Module 100](#_Toc417423665)

[4.2.21.1 Type 100](#_Toc417423666)

[4.2.21.2 Purpose 100](#_Toc417423667)

[4.2.21.3 Function 100](#_Toc417423668)

[4.2.21.4 Subordinates 100](#_Toc417423669)

[4.2.21.5 Dependencies 101](#_Toc417423670)

[4.2.21.6 Interfaces 101](#_Toc417423671)

[4.2.21.7 Resources 102](#_Toc417423672)

[4.2.21.8 References 102](#_Toc417423673)

[4.2.21.9 Data 102](#_Toc417423674)

[4.2.22 CropMask Trimming Data Preparation module 102](#_Toc417423675)

[4.2.22.1 Type 102](#_Toc417423676)

[4.2.22.2 Purpose 102](#_Toc417423677)

[4.2.22.3 Function 102](#_Toc417423678)

[4.2.22.4 Subordinates 103](#_Toc417423679)

[4.2.22.5 Dependencies 103](#_Toc417423680)

[4.2.22.6 Interfaces 103](#_Toc417423681)

[4.2.22.7 Resources 104](#_Toc417423682)

[4.2.22.8 References 104](#_Toc417423683)

[4.2.22.9 Data 104](#_Toc417423684)

[4.2.23 Reference Map Preparation module 104](#_Toc417423685)

[4.2.23.1 Type 104](#_Toc417423686)

[4.2.23.2 Purpose 104](#_Toc417423687)

[4.2.23.3 Function 104](#_Toc417423688)

[4.2.23.4 Subordinates 105](#_Toc417423689)

[4.2.23.5 Dependencies 105](#_Toc417423690)

[4.2.23.6 Interfaces 105](#_Toc417423691)

[4.2.23.7 Resources 105](#_Toc417423692)

[4.2.23.8 References 106](#_Toc417423693)

[4.2.23.9 Data 106](#_Toc417423694)

[4.2.24 Trimming module 106](#_Toc417423695)

[4.2.24.1 Type 106](#_Toc417423696)

[4.2.24.2 Purpose 106](#_Toc417423697)

[4.2.24.3 Function 106](#_Toc417423698)

[4.2.24.4 Subordinates 106](#_Toc417423699)

[4.2.24.5 Dependencies 106](#_Toc417423700)

[4.2.24.6 Interfaces 106](#_Toc417423701)

[4.2.24.7 Resources 107](#_Toc417423702)

[4.2.24.8 References 107](#_Toc417423703)

[4.2.24.9 Data 107](#_Toc417423704)

[4.2.25 Temporal resampling module 107](#_Toc417423705)

[4.2.25.1 Type 107](#_Toc417423706)

[4.2.25.2 Purpose 107](#_Toc417423707)

[4.2.25.3 Function 107](#_Toc417423708)

[4.2.25.4 Subordinates 108](#_Toc417423709)

[4.2.25.5 Dependencies 108](#_Toc417423710)

[4.2.25.6 Interfaces 108](#_Toc417423711)

[4.2.25.7 Resources 109](#_Toc417423712)

[4.2.25.8 References 109](#_Toc417423713)

[4.2.25.9 Data 109](#_Toc417423714)

[4.2.26 Crop mask post filtering module 109](#_Toc417423715)

[4.2.26.1 Type 109](#_Toc417423716)

[4.2.26.2 Purpose 109](#_Toc417423717)

[4.2.26.3 Function 109](#_Toc417423718)

[4.2.26.4 Subordinates 109](#_Toc417423719)

[4.2.26.5 Dependencies 110](#_Toc417423720)

[4.2.26.6 Interfaces 110](#_Toc417423721)

[4.2.26.7 Resources 110](#_Toc417423722)

[4.2.26.8 References 110](#_Toc417423723)

[4.2.26.9 Data 110](#_Toc417423724)

***List of figures***

[Figure 3‑1: Software System Architecture 22](#_Toc417422258)

[Figure 3‑2: Composite synthesis processor 27](#_Toc417422259)

[Figure 3‑3 : Temporal vegetation status indicator 28](#_Toc417422260)

[Figure 3‑4: Crop mask processor 28](#_Toc417422261)

[Figure 3‑5: Crop type processor 29](#_Toc417422262)

[Figure 3‑6 : Static architecture of the Sen2-Agri components (in blue the re-use components and green the new ones). 31](#_Toc417422263)

[Figure 3‑7 : Overview Activity Diagram of the System 32](#_Toc417422264)

[Figure 3‑8 : On-Demand Processing Activity Diagram 34](#_Toc417422265)

[Figure 3‑9 : Mock UI for System Process Configuration Utility 35](#_Toc417422266)

[Figure 3‑10 : Mock Layout and Content for Execution Dashboard 36](#_Toc417422267)

[Figure 4‑1 : Main Components and Their Relationships 39](#_Toc417422268)

[Figure 4‑2 : Simplified Orchestrator States 42](#_Toc417422269)

[Figure 4‑3 : Scheduler States 45](#_Toc417422270)

[Figure 4‑4 : DEM & WB Module Functional Diagram 54](#_Toc417422271)

[Figure 4‑5 : DEM L2 Computation 55](#_Toc417422272)

[Figure 4‑6 : DEM L2 Water Bodies Computation 56](#_Toc417422273)

[Figure 4‑7 : DEM L2 Water Bodies Computation (2) 56](#_Toc417422274)

[Figure 4‑8 : Bidirectional Correction Processor Diagram 64](#_Toc417422275)

[Figure 4‑9 : Compute Grid Angle Sub-Component Steps 66](#_Toc417422276)

[Figure 4‑10 : Iterative Composite Synthesis Processor Functional Diagram 69](#_Toc417422277)

[Figure 4‑11 : Compute Current Weight Sub-Component Logical Diagram 70](#_Toc417422278)

[Figure 4‑12 : Weight Update Sub-Component Diagram 71](#_Toc417422279)

[Figure 4‑13 : Flag Update Sub-Component Diagram 71](#_Toc417422280)

[Figure 4‑14 : Date Update Sub-Component Diagram 72](#_Toc417422281)

[Figure 4‑15 : Compute LAI Module Functional Diagram 75](#_Toc417422282)

[Figure 4‑16 : LAI Get Model Sub-Component Diagram 75](#_Toc417422283)

[Figure 4‑17 : Reprocess LAI Functional Diagram 77](#_Toc417422284)

[Figure 4‑18 : Functional diagram of the NDVI metrics module 79](#_Toc417422285)

[Figure 4‑19 : Samples Preparation Functional Diagram 82](#_Toc417422286)

[Figure 4‑20 : Temporal Feature Extraction Functional Diagram 86](#_Toc417422287)

[Figure 4‑21 : Learning Module Functional Diagram 89](#_Toc417422288)

[Figure 4‑22 : Classification and Validation Module Functional Diagram 92](#_Toc417422289)

[Figure 4‑23 : LUT Functional Diagram 94](#_Toc417422290)

[Figure 4‑24: Time Series Feature Extraction Module Functional Diagram 97](#_Toc417422291)

[Figure 4‑25: Extract bands at the right resolution functional diagram. 97](#_Toc417422292)

[Figure 4‑26 : Resampling Module Functional Diagram 100](#_Toc417422293)

[Figure 4‑27 : Temporal resampling functional module 108](#_Toc417422294)

***List of tables***

[Table 1‑1: Applicable documents 19](#_Toc417422295)

[Table 1‑2: Reference documents 19](#_Toc417422296)

[Table 1‑3: Acronyms 20](#_Toc417422297)

[Table 3‑1 : List of input parameters which can be defined by the user 24](#_Toc417422298)

[Table 3‑2 : List of indicators provided in Execution Dashboard 26](#_Toc417422299)

[Table 4‑1 : List of system components/modules 41](#_Toc417422300)

# Introduction

## Purpose and Scope

This document is the Design Definition File (DDF) of the Sentinel-2 for Agriculture (Sen2-Agri) project funded by the European Space Agency (ESA).

It describes the final algorithms of the 4 final products in the form of an Algorithm Theoretical Basis Document and provides a description of the general architectural and internal interfaces design of the Sen2-Agri processing system. It also defines specification of each component identified.

This DDF document is the key input of the other development activities of the task 3 as it defines the preliminary design and part of detail design of the system.

## Structure of the Document

After this introduction, this document contains 3 main sections that describe:

* The design overview;
* The detail description of system design in terms of architecture, components and interfaces;
* The traceability from the system requirements.

## References

### Applicable Documents

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **Title** | **Code** | **Issue** | **Date** |
|  | Sentinel-2 for Agriculture Statement of Work | EOEP-DUEP-EOPS-SW-13-0004 | 1.0 | 26/03/2013 |
|  | Sentinel-2 for Agriculture Technical Proposal |  | 1.0 |  |
|  | Sentinel-2 for Agriculture User Requirements Document | Sen2-Agri\_URD\_1.2 | 1.2 | 25/07/2014 |
|  | Sentinel-2 for Agriculture Product Specification Document | Sen2-Agri\_PSD\_1.0 | 1.0 | 29/07/2014 |
|  | Sentinel-2 for Agriculture Technical Specifications | Sen2-Agri\_TS\_1.0 | 1.0 | 29/07/2014 |
|  | Sentinel-2 for Agriculture Software Development Plan | Sen2-Agri\_SDP\_1.0 | 1.0 | 15/10/2014 |
|  | Sentinel-2 for Agriculture Test Data Set documentation | Sen2-Agri\_TDS\_1.2 | 1.2 | 27/02/2015 |
|  | Sentinel-2 for Agriculture Design Justification File | Sen2-Agri\_DJF\_1.1 | 1.1 | 15/04/2015 |

Table 1‑1: Applicable documents

### Reference Documents

|  |  |
| --- | --- |
| **ID** | **Title** |
|  | Algorithm Theoretical Basis Document for L3 monthly composite product |
|  | Algorithm Theoretical Basis Document for L3 biophysical product |
|  | Algorithm Theoretical Basis Document for L4 crop type product |
|  | Algorithm Theoretical Basis Document for L4 dynamic crop mask product |
|  | Orfeo ToolBox Cookbook, version 4.4 |
|  | Sentinel-2 User Handbook, 07/07/2014 Issue 1, Rev. 1 |
|  | LandSat8 L1 Data Format Control Book, Version 8 |
|  | SRTM Void Filled documentation, https://lta.cr.usgs.gov/SRTMVF |
|  | Sentinel-2 PSD, Rev 12 |
|  | User, installation and operating manual of MACCS chain, Issue 4, Rev. 3., 4/04/2014 |
|  | Gdal Utilities documentation, http://www.gdal.org/gdal\_utilities.html |
|  | Sentinel-1 Scientific Data Hub USerGuide, https://scihub.esa.int/twiki/do/view/SciHubUserGuide/WebHome |
|  | LandSat 8 Downlaod tool documentation, https://github.com/olivierhagolle/LANDSAT-Download |

Table 1‑2: Reference documents

### Acronyms and Abbreviations

|  |  |
| --- | --- |
| **Acronym** | **Definition** |
| AD | Applicable Document |
| ATD | Acceptance Test Document |
| CESBIO | Centre d'Études Spatiales de la Biosphère |
| CNES | Centre National d'Études Spatiales |
| CS | Communication & Systèmes |
| CS-R | CS Romania |
| DUE | Data User Element |
| ECSS | European Cooperation for Space Standardization |
| EO | Earth Observation |
| ESA | European Space Agency |
| GEO | Group on Earth Observation |
| GEOGLAM | GEO Global Agricultural Monitoring |
| GEOSS | GEO System of Systems |
| L1C | Level 1C |
| HMI | Human Machine Interface |
| OTB | Orfeo ToolBox |
| RD | Reference Document |
| PSD | Product Specification Document |
| RPM | Redhat Package Manager |
| S2 | Sentinel-2 |
| Sen2-Agri | Sentinel-2 for Agriculture |
| SoW | Statement of Work |
| SUM | Software User Manual |
| TDS | Test Data Set |
| TS | Technical Specifications |
| UCL | Université Catholique de Louvain |
| UR | User Requirement |
| URD | User Requirement Document |

Table 1‑3: Acronyms

# System Algorithm Theoretical Basis Documents

The objective of the Sen2-Agri system is to produce the different EO products required by the users based on the processing of the S2 L1C product. These different products are:

* composites of cloud free surface reflectance values ;
* dynamic cropland masks;
* cultivated crop type maps and area extent;
* vegetation status indicators.

A key activity of the project was the benchmarking which aimed at selecting the “best” algorithms for fulfilling to a maximum extent to the user requirements and the products specifications [AD.1].

For each product, a minimum of 5 concurrent algorithms were benchmarked – i.e. applied, assessed and inter-compared in considering the user-focus with critical attention – on a set of pre-defined test sites. A list of 12 sites was defined in the earlier stage of the project [AD.3] and EO and in-situ data were acquired for each of them, constituting the Test Data Set (TDS). This TDS is made of high spatial resolution time series from the year 2013 (images mainly acquired by Spot 4-Take 5 (S4-T5) and complemented with Landsat 8 (L8) and RapidEye (RE) sensors) and of field data from the same year.

The algorithms tested for each product were selected based on literature review and on an exploratory phase carried out to select the most interesting ones out of all possibilities offered by the state of the art.

In order to make this exercise as transparent as possible, the analytical approach for products assessment was agreed before the exercise.

The benchmarking resulted in the selection of one algorithm (or combination of algorithms) for each product. All the process is documented in 4 documents specific to each product [AD.8]. In the DDF, these selected algorithms are described in the form of Algorithm Theoretical Basis Document (ATBD).

The ATBD for the 4 products are presented as 4 separate chapters, one for each product:

* Algorithm Theoretical Basis Document for L3 monthly composite product [RD.1];
* Algorithm Theoretical Basis Document for L4 dynamic crop mask product [RD.4];
* Algorithm Theoretical Basis Document for L4 crop type product [RD.3];
* Algorithm Theoretical Basis Document for L3 biophysical product [RD.2].

Each report contains, at a minimum, the following content:

* An overview of the overall processing chain;
* For each processing step:
* The scope of the algorithm;
* The workflow with a clear identification of input and output data;
* The algorithm parameters;
* The pseudo-code representation.

# Design overview

## System static architecture

The system is composed of several layers:

* User interface layer
* Service layer
* Persistence layer

The architecture of the system as well as the relationship between the components identified is given in Figure 3 -1.

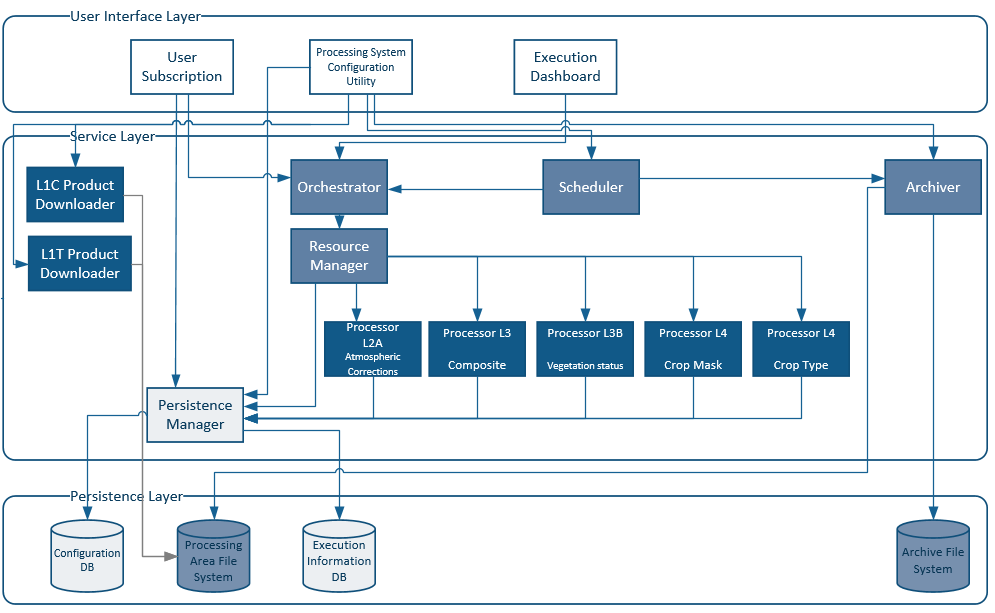


Figure 3‑1: Software System Architecture

### User Interface Layer

This layer groups together the components with which users (including system administrators) may interact. Depending on the user role, one or more of the following components may be available:

#### Processing System Configuration Utility

The configuration utility allows an administrator to change the processing facility parameters. System parameters are key-value pairs, with keys following a hierarchical scheme. Parameters can be configured with a global value that applies to all sites. If desired, a site-specific value can be assigned to some of them. The values have a type of integer, real, string, file or directory. The settings are stored in a database, but are accessed using the Persistence Manager service.

The utility uses metadata available in the database to generate its user interface. Some of the parameters can be marked as “advanced”, which will prevent the user from editing them. Advanced parameters can be modified by connecting directly to the database.

Besides the system-wide settings mentioned above, each product can be generated using different parameter values, usually be set by the user (e.g. when she adds a subscription). Editing the per-job settings is outside of scope of the configuration utility.

The list of user parameters is summarized in Table 3 -4.

| **Name** | **Role** | **Type** | **Mandatory** | **Advanced** |
| --- | --- | --- | --- | --- |
| User Area of Interest (AOI) | define the area covered by the Sen2-Agri Product | File  Format: shp file | Yes | No |
| Start of the Period  End of the Period | define the season monitors by the Sen2-Agri System | Date  Date | Yes  Yes | No |
| Sensor Configuration | define if the system should use only Sentinel-2 or also LandSat8 data | Boolean  Default Value is S2 only | Yes | No |
| Reference samples for crop type | define the reference samples used by the crop type | File  Format: shp file | No[[1]](#footnote-2) | No |
| Reference samples for crop mask | define the reference samples used by the crop mask | File  Format: shp file | No | No |
| Time Resampling step | define the time resampling step used in the temporal resampling | Integer  Default value: 5 days | No | Yes |
| Size of the Composite Window | define the duration of the period used for the composite synthesis | Integer  Default value: 40 days  Min-Max value: [30 -50] days | No | No |
| Integration period for biophysical vegetation indicator | define the duration of the period used for the biophysical product | Integer  Default Value: 0 day  Min-Max value: [0,3] days | No | No |
| DEM auxiliary data | define the location of the user DEM | Directory  Format: USGS Void Filled 30m Product  Default value: Empty | No | Yes |
| User LandCover map | define the location of a previous landcover map from which a binary crop-no crop mask could be derived | File  Format: Raster, GeoTiff  Default value: empty | No | Yes |
| User LandCover nomenclature | define the location of the nomenclature associated to the | File  Format: a table with for each label if it is crop or no crop, TXT  Default value: empty | Yes if the User LandCover map is set  No otherwise | Yes |
| User stratification areas | define the different ecological areas available in the AOI of the user | File  Format: polygon with unique ID, shp file  Default value: empty | No | Yes |
| Maximum L2a/L3a/L3b/L4a/L4b age before archival | Define the time after products are archived. The archive is considered to be always available, but is located on slower storage. | Integer  Default value: 5 days | Yes | Yes |
| Maximum L2a/L3a/L3b/L4a/L4b retention time | Define the time after products are deleted from the archive. | Integer  Default value: never | Yes | Yes |
| Intermediate product retention time | Define the time after intermediate products are deleted.  Note: Temporary files generated and used by the processors are removed when they are no longer needed. | Integer:  Default value: 5 days | Yes | Yes |

Table 3‑4 : List of input parameters which can be defined by the user

#### Execution Dashboard

The Execution Dashboard allows to monitor at any time what is executing in the system (i.e. what processing chains – or processors), but also to see the system resources utilisation. Due to execution information persistence, it also may allow to see historical series of execution data.

The user interface will allow monitoring the system from a system point of view but also from a user point of view. Table 3 -5 presents the different indicators provided in this dashboard.

| **Name** | **Purpose** | **Type** | **Product/System** | **Granularity** |
| --- | --- | --- | --- | --- |
| General status of the system | Indicate the global status of the system | Three status: run, not run, block | System | Global |
| L2A processor status | Indicate status of the L2A processor | Three status: run, not run, block | System | Global |
| L3A processor status | Indicate status of the L3A processor | Three status: run, not run, block | System | Global |
| L3B processor status | Indicate status of the L3B processor | Three status: run, not run, block | System | Global |
| L4A processor status | Indicate status of the L4A processor | Three status: run, not run, block | System | Global |
| L4B processor status | Indicate status of the L4B processor | Three status: run, not run, block | System | Global |
| General User Configuration | Report the information set by the users at the initialisation | cf. the Table 3 -4 | System | Global |
| Mean tile processing time by processors | Indicate the mean processing time for each processor | Time | System | Processor |
| Data volume status | Indicate the disk usage and its overall capacity | Integer  Unit: MiB | System | Global |
| Overall tile status | Indicate how many tiles have been processed with errors | Integer | System | Global |
| L2A Cloud cover | Indicate the cloud cover of each L2A tile produced | Float | Product | Tile |
| L2A Mean AOT value | Indicate the mean AOT of each L2A tile produced | Float | Product | Tile |
| L3A Invalid pixel percentage | Indicate the invalid percentage of each L3A tile | Float | Product | Tile |
| L3B error mean and variance | Indicate the error evaluate during the LAI retrieval of each tile | Float  Float | Product | Tile |
| L4A Overall Accurary and FScore | Indicate the OA and FScore of each L4A tile | Float  Float | Product | Tile |
| L4B Overall Accurary and FScore | Indicate the OA and FScore of each L4B tile | Float  Float | Product | Tile |

Table 3‑5 : List of indicators provided in Execution Dashboard

The dashboard is implemented as a desktop application that is accessible to the system administrator. Information is retrieved from:

* The Execution Information database
* The Orchestrator
* The local system

#### User Subscription Manager

This component handles the configuration and launch of on-demand processing by a regular user. It also allows a user to express his/her interest in certain products that are to be produced by the system processors, so that, when they will be available, the user may download them.

### Service Layer

This layer implements almost all the logic of the processing facility. The main components of are:

#### L1C Product Downloader

It will connect to the ESA L1C Product data storage and will fetch the available products into the processing area file system.

#### L1T Product Downloader

It will connect to the USGS L1T Product data storage and will fetch the available products into the processing area file system.

#### The Orchestrator

The Orchestrator is the main component which is used to manage the Sen2-Agri software components on the system. Its role is to monitor the occurrence of new files in the processing area file system, determine the processing chain to be launched, handle user requests and monitor the execution of the current processing chains.

#### Scheduler

This component is responsible with the execution (i.e. Orchestrator invocation) of timed jobs. A timed job is an execution unit that needs a certain quantity of input information that may not be available when the job is defined.

There are two categories of timed jobs:

* Jobs that should execute repetitively with a given period (for example, each 30 days);
* Jobs that should execute when the necessary input information becomes available (for example, when a number of L1C products for the same site are received).

The Scheduler communicates with the Orchestrator using the IPC API of the latter component.

#### Resource Manager

The Resource Manager will be in charge with the parallel execution of different jobs (execution of processors with a given set of parameters). In the centre of this component will be SLURM workload manager.

#### Processors

The processors will be a collection of one or several modules dedicated to produce at tile level one of the Sen2-Agri products. These processors are split into several modules (see section 4.2) to allow different parallelization strategies. The processors will be:

* Atmospheric corrections processor that will produce L2A products
* Temporal synthesis processor that will deliver the L3A products (Figure 3 -2)

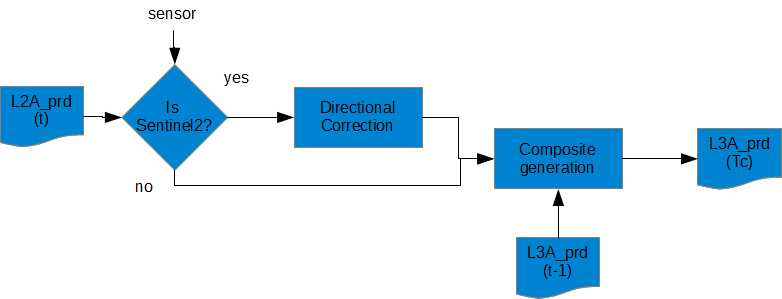


Figure 3‑2: Composite synthesis processor

* A biophysical vegetation status processor (Figure 3 -3)

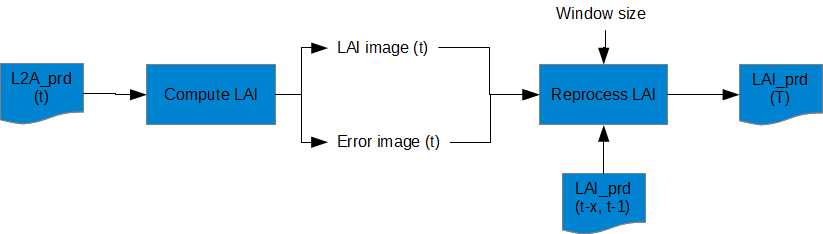


Figure 3‑3 : Temporal vegetation status indicator

* Crop mask processor (Figure 3 -4)

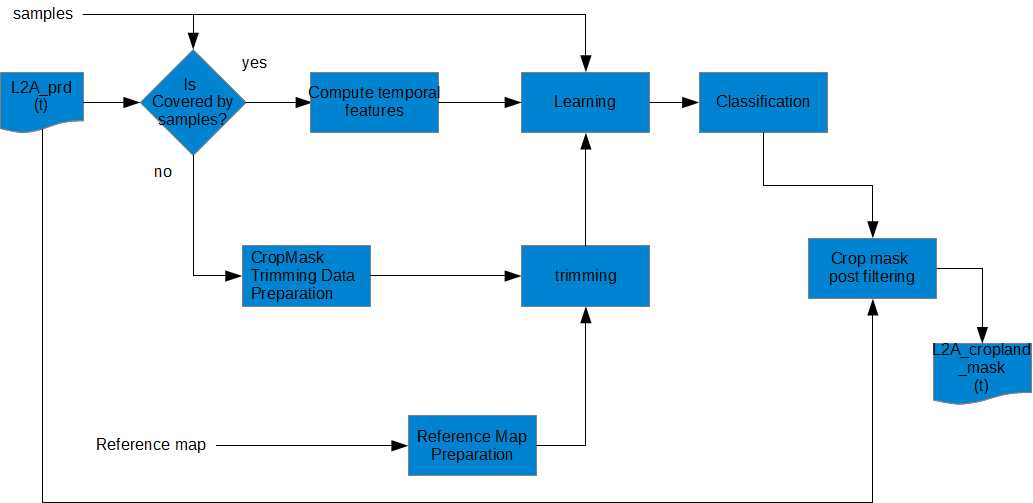


Figure 3‑4: Crop mask processor

* Crop type processor (Figure 3 -5 )

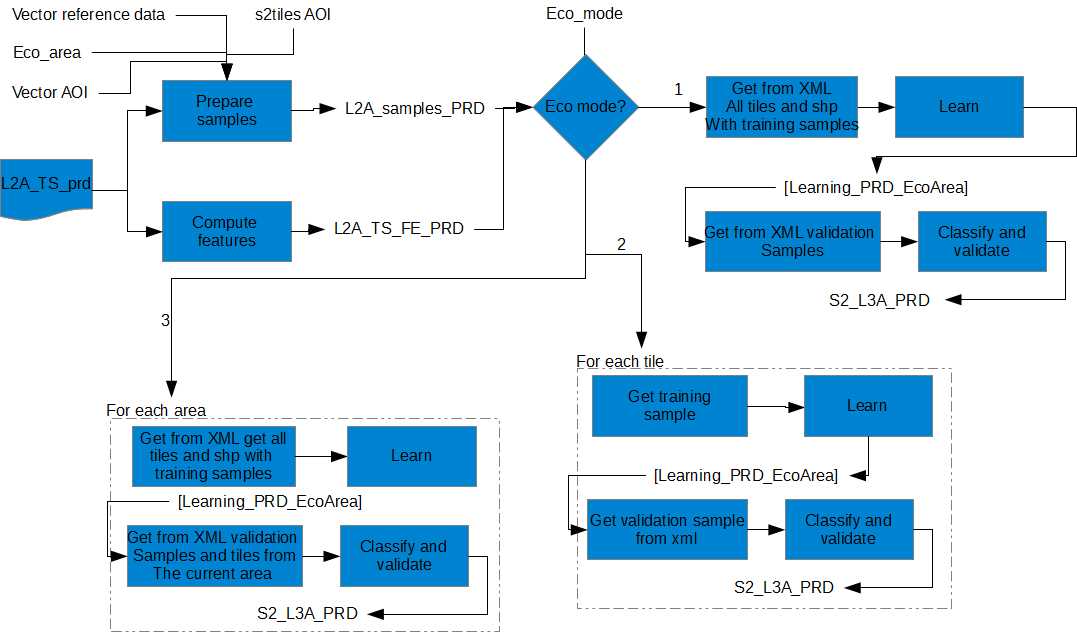


Figure 3‑5: Crop type processor

#### Modules

The modules will be actually wrappers over a set of executables and functions that can be invoked with parameters. The list of modules is summarized in Table 4 -6.

The wrappers will have also the role to collect information about the execution (name of the input and output product, start time and execution duration etc.) and to send them to the persistence manager in order to be saved to the Execution information database.

#### Archiver

This service handles the retention policies for final / intermediate / user products.

For each product type, a retention policy can be defined. This will consist in specifying a time duration during which the product(s) can be downloaded or used.

#### Persistence Manager

The Persistence Manager will be the interface between the other modules of the system with the persistence layer databases. It stores the execution information like the start time, duration, executed processor, input and output files to the Execution Information Database or the information needed for handling the user requests (input product names, area of interest etc.)

### Persistence Layer

#### Configuration Database

This database will contain the following information:

* Configuration parameters of the system
* Configuration for the user subscriptions

#### Execution information database

The Execution Information database will store information about executions (start time, duration etc.), about the input intermediate and resulted products (name, size of the product etc.), processors invoked. It also can store information about the intermediate states when there is a multi-step processing that can be spanned in time, depending on the availability of the products needed at the current step in order to continue processing.

#### Processing Area (File System)

As the new products are retrieved by the L1C downloader, they will be stored in the processing area file system. The orchestrator will monitor for specific changes the folders where the new products are added and will trigger the processing of these products.

During processing, the intermediate files will be also stored in this file system as well as the final products. The final products might be also input for other processors.

#### Archive (File System)

According to the defined retention policy, the files that will fall into this policy will be moved from the Processing Area file system to the Archive file system.

### System static package and library architecture

The Sen2-Agri architecture has been designed to maximise re-use of existing components: open-source component as OTB and GDAL or components developed during the benchmarking phase. Figure 3 -6 shows that the Sen2-Agri modules have designed as wrappers around

* Existing functionalities from GDAL/OGR library or tools to deal with vector data and OTB applications to deal with image processing.
* New image processing functionalities available in modules through a new set of applications: Sen2-Agri ones.

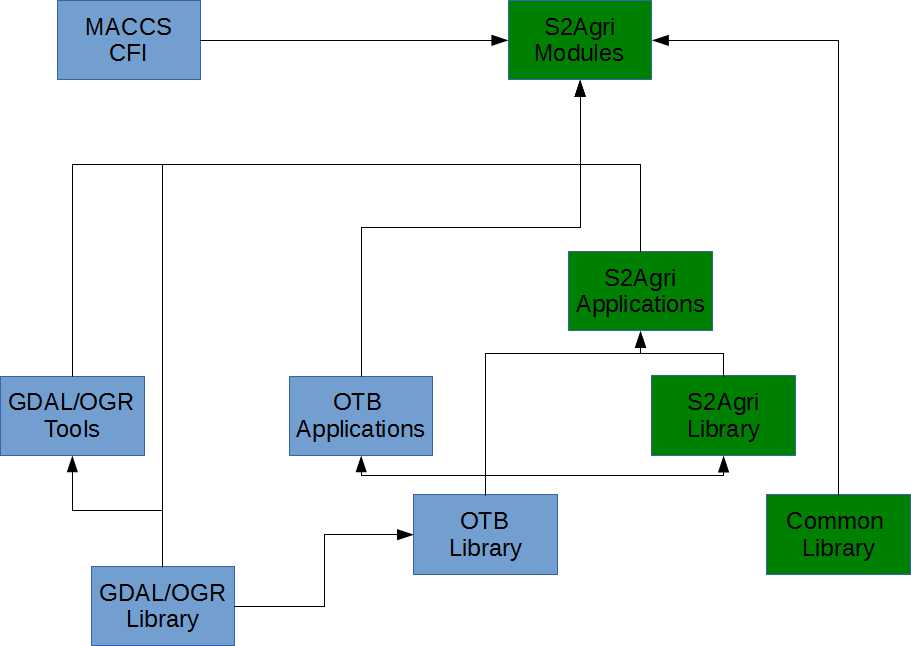


Figure 3‑6 : Static architecture of the Sen2-Agri components (in blue the re-use components and green the new ones).

## System Dynamic Architecture

Figure 3 -7 presents a simplified activity diagram.



Figure 3‑7 : Overview Activity Diagram of the System

During normal operation, the orchestrator will perform the following verifications:

* If there is a scheduled task to be executed. A typical scheduled task is the archiving task. In this case the archiver will be invoked in order to execute the defined retention rules.
* If there is a new product available in the processing area file system. In this case, according to the type of the detected product will be determined the processing chain to be invoked.
* If there is a new user request. In this case, the orchestrator will read the configuration parameters from the database for this new request and will create a new job to be executed by the resource manager.

Once a new job is added to the resource manager’s queue, the processor wrapper from the job will be invoked with the specified parameters. All the relevant information will be saved by the processor wrapper into the execution information database via the persistence manager.

If an error occurred during processing, the problem is logged and the orchestrator is informed. Depending on the error type, the orchestrator might decide whether the processing of the failed job should be re-executed (for example in the case of no more disk space, it might be first performed a spontaneous launch of the archiver before trying re-execution of the processing).

## System Behaviour

### On-Demand Processing

Figure 3 -8 presents the flow when the user performs a processing request.



Figure 3‑8 : On-Demand Processing Activity Diagram

## Interfaces Context

### ESA Sentinel-2 Interface

Currently the official ESA Sentinel-2 download interface is not known. The NgEO interface is not available at this moment. Therefore we will consider an alternative solution based on the existing Sentinel-1 data hub interface and it’s API. Due to the fact that the official Sentinel-2 data hub API is not currently know, the L1C product downloader will be considered as a prototype until the final technical solution will be available at ESA side.

### USGS LandSat8 Interface

Currently no official USGS LandSat8 download interface is available. Several solutions are possible: firstly an unofficial API found by Olivier and secondly an API based on a Google-USGS mirror. During the PDR, it has been decided to support only the first option. Therefore, the L1T product downloader will be based on this code.

### User Interface

Some mock samples of the user interface components are presented in Figure 3 -8Figure 3 -9 and Figure 3 -10.

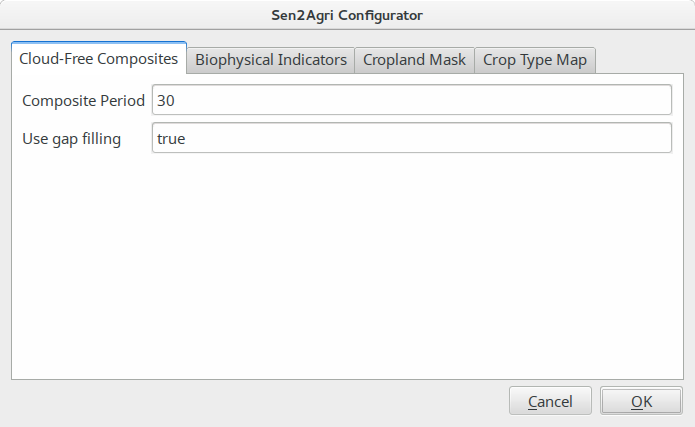


Figure 3‑9 : Mock UI for System Process Configuration Utility



Figure 3‑10 : Mock Layout and Content for Execution Dashboard

### Orchestrator Interface

The interface exposed by the Orchestrator service allows the system to be controllable from within other systems. The reference client implementation for it, in the scope of the current project, is represented by the Configuration Utility and Execution Dashboard.

### Sentinel-2 Toolbox Interface

For the integration of Sen2-Agri processors in the Sentinel-2 Toolbox, no special consideration should be made since a processor is an independent OTB application. This integration is controlled by Sentinel-2 Toolbox via its Standalone Tool Adapter module. However, certain aspects have to be considered when integrating a Sen2-Agri processor:

* If there are specific pre-requisites for a processor, they should be addressed in the pre-execution template of the STA module definition;
* The command line parameters of the processor have to be described in the command line template of the STA module definition;
* Any post-processing actions (eg. cleaning up the temporary files) should be addressed in the post-execution template of the STA module definition.

## Specific constraint

### SLURM

SLURM allocates exclusive and/or non-exclusive access to resources (computer nodes) to users for some duration of time so they can perform work. It also provides a framework for starting, executing, and monitoring work (typically a parallel job) on a set of allocated nodes. Finally, it arbitrates contention for resources by managing a queue of pending work. Even this tools is mainly used for cluster task management, it is also very useful on a single platform with multiple cores.

The SLURM multifactor priority plugin will be used to assign different priorities for system (Sen2-Agri internal) and user-requested jobs. This is accomplished by setting the Quality of Service factor according to the job type.

SLURM also allows for currently executing and pending jobs to be cancelled. Using this interface, the Orchestrator will allow the system administrator to gracefully stop processing when it is needed.

SLURM will be in charge with the parallel execution of different jobs from the resource manager from the orchestrator.

### D-Bus

D-Bus is a protocol for local inter‑process communication and can handle authorization and introspection. Client and server D‑Bus libraries are available for the common programming languages, together with command‑line tools for discovering services and performing calls.

D-Bus objects will be used by the Orchestrator API for the inter-process communication.

### Cron

Cron is a time-based job scheduler in Unix-like computer operating systems. It is used to schedule jobs (commands or shell scripts) to run periodically at fixed times, dates, or intervals. It will be used for triggering jobs like archiving of old products or clean up old user generated products.

## Memory and CPU Budget

In order to make system and user processes as non-competitive as possible in terms of system memory and CPU, we introduce a partitioning of resources. Thus, in a nominal load of the system, assuming there are both system processes and user processes to be executed at the same time:

* 67% of CPU is allotted to system processes (i.e. 4 physical cores / 8 logical processors)
* 33% of CPU is allotted to user processes (i.e. 2 physical cores / 4 logical processors)
* 16 GB of system memory is allotted to system processes (50% of total system memory)
* 8 GB of system memory is allotted to user processes (25% of total system memory)

System processes have priority over the user ones. Therefore, if the system processes need more CPU, the above ratios can change, but not under a specific threshold.

## Design Standards, Conventions and Procedures

The SDP [AD.6] summarizes all the elements related to this section.

# System Design

## Overall Architecture

Figure 4 -11 presents the relationship between the main modules of the application.



Figure 4‑11 : Main Components and Their Relationships

The core of the system will be the orchestrator. It will be able to:

* Schedule a task and receive notifications when a task timer elapsed or a new product arrived
* Start, stop or schedule a processor
* Inspect a processor state and save its execution info to the database (via the persistence manager).
* Read and save configurations to the database via the persistence manager.
* Checks the execution state of the running processors and takes the required actions (to restart the execution of the processor if a recoverable error occurred or to stop it on severe error).

## Components Design

Table 4 -6 summarizes the list of main components/modules and their granularity/parallelism strategy.

| **Name** | **Granularity** | **Section** |
| --- | --- | --- |
| Orchestrator | N/A | 4.2.1 |
| Scheduler | N/A | 4.2.2 |
| L1C Product Downloader | N/A | 4.2.3 |
| L1T Product Downloader | N/A | 4.2.4 |
| Compute Tile list from AOI module | N/A | 4.2.5 |
| DEM&WB L2 Module | by S2 and L8 tiles | 4.2.6 |
| L2A PreProcessing Module | by S2 and L8 tiles | 4.2.8 |
| L2A Processing Module | by S2 and L8 tiles | 4.2.7 |
| L2A PostProcessing Module | by S2 and L8 tiles | 4.2.9 |
| Bidirectionnal Filtering Module | by S2 tiles | 4.2.10 |
| Iterative Composite Synthesis Module | by S2 tiles | 4.2.11 |
| Compute LAI module | by S2 tiles | 4.2.12 |
| Reprocess LAI module | by S2 tiles | 4.2.13 |
| NDVI metrics module | by S2 tiles | 4.2.14 |
| Samples Preparation Module | N/A | 4.2.15 |
| Compute Temporal features Module | by S2 tiles | 4.2.16 |
| Learning Module | by S2 tiles or group of tiles | 4.2.17 |
| Classification and Validation Module | by S2 tiles | 4.2.18 |
| L4B-LUT Generation Module | all tiles | 4.2.19 |
| Time series Feature Extraction module | by S2 tiles | 4.2.20 |
| Resampling module | by S2 tiles | 4.2.21 |
| CropMask Trimming data preparation module | by S2 tiles | 4.2.22 |
| Reference map preparation module | by S2 tiles | 4.2.23 |
| Trimming module | by S2 tiles | 4.2.24 |
| Temporal resampling module | by S2 tiles | 4.2.25 |
| Crop mask post filtering module | by S2 tiles | 4.2.26 |

Table 4‑6 : List of system components/modules

### Orchestrator Component

#### Type

The orchestrator is a module for sequencing and executing tasks and dependencies in maximum concurrency. Its function will be to handle new product occurrence, handle user requests, monitor executing processors and decide the new step to be executed in the processing chain.

#### Purpose

Several types of operations can occur in the same time like the user requests, new products to be processed or archiving operations. All these operations need to be handled concurrently in a uniform manner. The orchestrator purpose is to dispatch and monitor all the task executions between various components.

#### Function

The main functionalities of the orchestrator are:

* Monitors the processing area file system in order to determine if new products are available (either retrieved by the L1C downloader or products obtained after an intermediate processing step) and determines their types and the processing chain that needs to be applied further.
* Executes periodic tasks for processing products.
* Handles the user requests for launching a new processing chain
* According to the input data (new files, user input) decides what processor should be invoked and its parameters and adds new jobs containing these information to be executed by the Resource manager.
* Monitors the status and progress of a certain execution chain (compute the next state to be executed, check if the execution finished or is blocked).
* Can request to the Resource Manager to stop the execution for a certain processing job.

A simplified state diagram for the orchestrator is shown in Figure 4 -12.



Figure 4‑12 : Simplified Orchestrator States

The Orchestrator offers an IPC (Inter-Process Communication) API (Application Programming Interface), which other components use to raise notifications or query the system status. This interface is also used by the system administrator when sending commands.

The Orchestrator API takes the form of a D-Bus object.

#### Subordinates

Scheduler, Resource Manager, Processors

#### Dependencies

The orchestrator is the first component to be executed in the system. The persistence layer components should be initialized before it is created (the Configuration database, the processing area file system, the execution information database and the archive file system).

#### Interfaces

##### startProcessor

Creates a Task component with an instance of a processor, from the internal list of registered processors, given the input parameters, and dispatches the newly created component to the Resource Manager to be invoked with properly allocated resources (CPU and memory).

The method should be asynchronous since the Orchestrator should not be blocked by waiting for the initialisation of the new process.

Input:

* Processor: the processor component instance;
* Parameter []: the list of processor parameters.

Output:

* TBD

##### stopProcessor

Instructs the Resource Manager to stop a task process. If the latter succeeds, the task will be removed from the internal execution queue.

The method should be asynchronous since the Orchestrator should not be blocked by waiting for the task process closure.

Input:

* Processor instance

Output:

* None

##### scheduleProcessor

Creates a Task component with an instance of a processor, from the internal list of registered processors, given the input parameters, and dispatches the newly created component to the Scheduler, so that its invocation can be handled later.

The method can be synchronous since its execution is short.

Input:

* Processor: the processor component instance;
* Parameter []: the list of processor parameters.

Output:

* TBD

##### inspectProcessor

Requests the available execution information (i.e. performance counters) of a processor process (like, for example, the consumed CPU time, memory, etc.).

Input:

* Processor: the processor to be queried;

Output:

* The collection of performance counters values.

##### readConfiguration

Reads from Configuration Database all the parameters of all processors of all scheduled tasks.

##### saveConfiguration

Persists all the parameters (of both processors and scheduled tasks) to the Configuration Database.

##### saveExecutionInfo

Persists the execution information (ProcessorState) received from a processor (by invoking first ***inspectProcessor*** function.

##### onProcessorFinished

This call-back method is invoked by a processor component when it finishes its execution (either successfully or with an error).

#### Resources

Not applicable here

#### References

Not Applicable

#### Data

##### Registered Processors

The component maintains a list of registered processors so that it can easily create new instances of such processor components.

##### Execution Queue

The component maintains a priority queue for executing tasks.

### Scheduler Component

#### Type

This component is in charge of executing tasks that are periodic or need to be executed at a certain moment of time or tasks that need to be triggered based on a certain event.

The scheduler component is composed of two parts:

* An independent component that is executed using the time-based job scheduler of the operating system (for example cron jobs in Linux). A typical job for this component will be invoking the archiver module in order to run the product retention rules or to clean up the products during the user requests.
* A component that is integrated in the Orchestrator and that is intended for monitoring asynchronous external events (for example the notification that a new product is available in the processing area file system) or for monitoring the executing tasks (for example, if a certain task takes too long it should be stopped).

#### Purpose

Certain executions need to be run periodically at fixed times, dates, or intervals. The component is needed in order to provide timed based notifications when a certain execution have to take place.

#### Function

The scheduler will trigger, based on the current timeout if a task need to be created (either an archiving task or a product processing task. If it is the case, it will send the created task to the Orchestrator in order to be executed.



Figure 4‑13 : Scheduler States

#### Subordinates

Not applicable.

#### Dependencies

Not applicable.

#### Interfaces

##### scheduleTask

Adds a new task in the internal list of the scheduler.

This method will be a non-blocking.

Input:

* Task: The task containing the processor and the parameters.

##### onTaksTimerElapsed

This is a call-back function that is called by the scheduler when the timeout for a certain task expired. In this call-back, the orchestrator is informed about this event by providing the processor and the parameters encapsulated in the task.

Input:

* Task: The task whose timeout expired.

##### onNewProductArived

This is a call-back function that is called by the scheduler when a new product is detected. In this call-back, the name of the product is sent to the orchestrator.

#### Resources

Not applicable.

#### References

Not applicable.

#### Data

The scheduler will manage internally a list of tasks. If a timeout is detected for one of the task in the list, then an event will be triggered to the orchestrator for it. Also, the folders for monitoring the occurrence of new products are internally kept by the scheduler.

### L1C Product Downloader

#### Type

This component is a part of the service layer of the Sen2-Agri System.

#### Purpose

The module will be in charge of periodically downloading the available Level-1 S2 products regarding a defined area of interest. It is placed before the data-driven processing in the global workflow.

#### Function

The module is designed as a daemon which is activated periodically. In order to retrieve the data soon after its release, the frequency at which the module is activated will be critic parameter.

The execution of the module is decomposed in the following step:

At each activation:

* The module request the list of available product according to the area of interest list and a time window,
* Each product from the retrieved list is checked for a previous download,
* Each product from the retrieved list that has not been previously downloaded is queued and downloaded as soon as a download slot is available,
* Once downloaded, each product integrity is check with available methods.

#### Subordinates

According to the previous description, each sub-component of this module could be described in more details.

##### Daemon

The module is designed as a daemon. It runs in background and activates the product request in a scheduled manner. For example, it activates every 8 hours the request for each area of interest. It also manages the execution of download task according to the availability of a download slot.

A possible is to use the “cron” scheduler to orchestrate the activation of the module at a defined frequency:

Example of a cron configuration to run a script every 2 hours:

0 \*/2 \* \* \* /path/to/s2a\_dhusget.sh

The first step is the retrieval of the product list of every published product over an area of interest during the last N hours. This list is provided by the data provider server after a request sent by the download module. This part of the module implementation will be based on the dhusget script available through the S-1 Scientific Data Hub user guide. The parameters will be extracted from the auxiliary files.

dhusget.sh [-d <DHuS URL>] [-u <username> ] [ -p <password>]

[-t <time to search (hours)>]

[-c <coordinates ie: x1,y1;x2,y2>]

[-T <product type>] [-o <option>]

NB: The dhusget script includes both the server request to retrieve the product list and the sequential download of every product from the retrieved list. In order to provide more control over the download process, these steps will be separated in two different submodules: the product of interest list retriever and the product download sub-modules.

The daemon output is the resulting product list. This list will then be filtered.

##### History Check

To prevent multiple downloads of the same product, the retrieved product list will be filtered to remove the already downloaded products. Each product successfully passing this filtering process will be added in the download queue.

This filtering step takes as input the product list and the history database associated to the corresponding area of interest and output an updated download queue.

##### Product Download

The product download process is composed of:

* a download queue
* a data retriever

The download queue is used to store the product to be downloaded and consists in a database of product name and populated by the history check processing step. The product status in the queue database is updated by the process according to the state it is in (pending, in progress...). It allows to store pending product downloads in case the number of available download slots is not sufficient to start the download of every selected products.

The Download tasks will be executed using a modified version of the dhusget script available through the S-1 Scientific Data Hub user guide. It will be run by the daemon itself given the availability of a download slot and a pending product download in the queue.

The dhusget script input parameters will be extracted from the auxiliary files.

dhusget.sh [-d <DHuS URL>] [-u <username> ] [ -p <password>]

[-t <time to search (hours)>]

[-c <coordinates ie: x1,y1;x2,y2>]

[-T <product type>] [-o <option>]

Once again, it is important to notice that the product download sub-module will be base on a part of the dhusget script.

Using the daemon capabilities of the download module, the download queue is watched to start a download as soon as both a download slot and a queued download are detected. The availability of a slot is determined by the difference between the user defined number of authorized slot and the number of product in the queue with the status “in progress”.

##### Integrity check

Once both the product and manifest are finished, the integrity check sub-module is executed on the product.

For each file in the product, the integrity check will be provided by controlling the MD5 checksum embedded in the associated product manifest. Eventually, the integrity check will be limited to relevant subset of product files.

For each file to be validated:

* compute the file MD5 checksum

md5sum [OPTION]... [FILE]...

* retrieve checksum from the manifest
* compare and:
  + reject product if different and re-add it to the download queue,
  + move to a dedicated repository if validated and alter the download module history database accordingly.

The command line tool md5sum allows to compute file checksums and is widely used. The validation checksum can be retrieve from the manifest using regular expressions and standard command line tools like grep and sed.

Nevertheless, the use of XML toolkits like XMLStarlet can reduce the potential complexity of using grep/sed command line tools to retrieve the validation checksum from the product manifest.

The described product integrity validation implementation provides a small footprint solution in terms of dedicated software and dependencies.

#### Dependencies

The dependences of this module are the following:

* The script provided by ESA to retrieve data from DataHub
* Common library to provide capabilities to check a L1C product

All these components should be initialized at the beginning of the module.

#### Interfaces

The main interfaces in term of data flow of its module are:

* As input, the list of L1C tiles (name and bounding box) which covered the Area of Interest of the user
* As output, downloaded L1C products will be stored in a sub-directory,
* As auxiliary data: a global configuration file including the URL of the remote data provider and the associated login information will be available as an auxiliary file.
* As image processing parameters: Not Applicable here

#### Resources

This module requires having an internet connexion.

#### References

It is assumed that level 1 S2 products dissemination will be based on a Open Data Protocol equivalent to the API used by the “Sentinel-1 Scientific Data Hub” [RD.12].

#### Data

Not Applicable here.

### L1T Product Downloader

#### Type

This component is a part of the service layer of the Sen2-Agri System.

#### Purpose

The module will be in charge of periodically downloading the available L1T LandSat8 products regarding a defined area of interest. It is placed before the data-driven processing in the global workflow.

#### Function

The module is designed as a daemon which is activated periodically. In order to retrieve the data soon after its release, the frequency at which the module is activated will be critic parameter.

The execution of the module is decomposed in the following step:

At each activation:

* The module request the list of available product according to the area of interest list and a time window,
* Each product from the retrieved list is checked for a previous download,
* Each product from the retrieved list that has not been previously downloaded is queued and downloaded as soon as a download slot is available,
* Once downloaded, the integrity of each product is check with available methods.

#### Subordinates

According to the previous description, each sub-component of this module could be described in more details.

##### Daemon

As for the L1C product downloader, a daemon executes the task regularly.

##### History Check

To prevent multiple downloads of the same product, the retrieved product list will be filtered to remove the already downloaded products. Each product successfully passing this filtering process will be added in the download queue.

##### Product Download

The downloading operation is executed through the python code provided by Olivier to the community. All the related information is available at [RD.13]

##### Integrity Check

Once both the product and manifest are finished, the integrity check sub-module is executed on the product.

For each file in the product, the integrity check will be provided by controlling the MD5 checksum embedded in the LandSat8 product.

For each file to be validated:

* compute the file MD5 checksum

md5sum [OPTION]... [FILE]...

* retrieve checksum from the metadata
* compare and:
  + reject product if different and re-add it to the download queue,
  + move to a dedicated repository if validated and alter the download module history database accordingly.

#### Dependencies

The dependences of this module are the following:

* The python code available from the Olivier GitHub
* Common library to provide capabilities to check a LandSat8 L1T product

All these components should be initialized at the beginning of the module.

#### Interfaces

The main interfaces in term of data flow of its module are:

* As input, the list of LandSat8 L1T tiles (name and bounding box) which covered the Area of Interest of the user
* As output, downloaded LandSat8 L1T products which are stored in a sub-directory,
* As auxiliary data: a global configuration file including the URL of the remote data provider and the associated login information will be available as an auxiliary file.
* As image processing parameters: not applicable here

#### Resources

This module requires having an internet connexion.

#### References

No official public API exists to download LandSat8 data from USGS server.

#### Data

Not Applicable here.

### Compute Tile Lists From AOI Module

#### Type

The compute tile list from user AOI is a module which provides to the system the first information about the number of S2 tiles or L8 tiles covered and their temporal distribution. This component can be also exposed to the user to allow him to evaluate the spatial temporal coverage of its request.

#### Purpose

This module converts the user AOI and monitoring period in term of L1C and L1T tile information: Tile ID, orbits, planned acquisition date.

#### Function

This module is based on the following pseudo-code:

For each S2 or L8 tiles which intersects the AOI

Retrieve the related orbits (only one for L8 and several for S2)

Save the tile\_ID, its extent, the orbits number.

Based on the orbits days, the first acquisition date, the availability of S2B and the orbit cycle duration, compute the temporal distribution of the acquisition.

#### Subordinates

Due to the simplicity of the code and the fact that all the operations can be done easily in Python and with the OGR API, no subordinates are presented here.

#### Dependencies

The dependence of this module is the following:

* Common library/package to provide capabilities to read and write a vector data file.

This component should be initialized at the beginning of the module.

#### Interfaces

The main interfaces in term of data flow of its module are:

* As input :
  + A vector data file containing all world s2 tiles extent
  + A file including all S2 orbits
  + A vector data file containing all world L8 tiles extent and their orbits
* As output :
  + A vector data file containing all tiles extent concerned by the user AOI and monitoring period
  + A XML file predicting next date each tile will be covered
* As auxiliary data
  + Not applicable here
* As processing parameters
  + in case of Sentinel2, S2B availability

#### Resources

Not Applicable

#### References

The module is strongly related to the Sentinel-2 User guide [RD.6] and the LandSat8 one [RD.7].

#### Data

Not applicable here

### DEM&WB Processing Module

#### Type

The DEM and Water Bodies is a module which represents a pre-processing step of the L2A processor. This module is a collection of executable or internal functions.

#### Purpose

DEM and Water Bodies module allows to get a DEM product at level L2. This DEM L2 product contains:

* a DEM at the given r1 resolution,
* a DEM at the given coarse resolution
* a DEM at resolution r2 for Sentinel 2 products
* slope at r1 resolution (and r2 for Sentinel 2 products)
* aspect at r1 resolution (and r2 for Sentinel 2 products)
* water mask at coarse resolution

The DEM module consists in building a DEM to cover the given extent, computing aspect and slope, and save all these files at different resolutions. The water bodies module crop the given water body and save it at the given resolution.

#### Function

This module is composed by three sub modules as shown in Figure 4 -14.

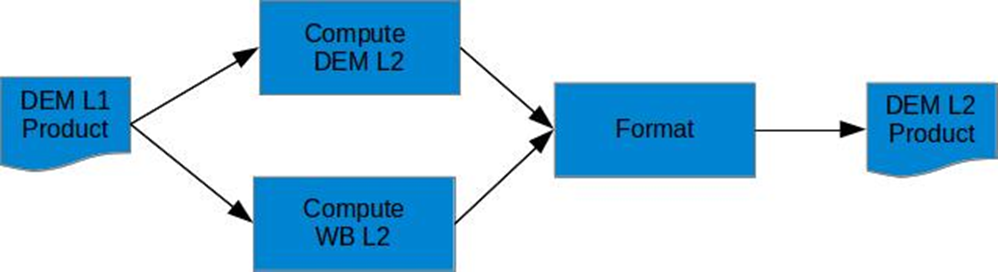


Figure 4‑14 : DEM & WB Module Functional Diagram

#### Subordinates

Based on Figure 4 -14, the list of sub-components is the following:

* Compute L2 DEM: this sub-component is below
* Compute L2 water body: this sub-component is below
* Format: create a hdr file which describes the DEM product and produce an archive

##### Compute L2 DEM

This component is divided in two main steps: the DEM computing at r1, r2 and coarse resolution, and the generation of the aspect and the slope at r1 and r2 resolution. Figure 4 -15 presents the diagram to get DEM, for the right resolutions.

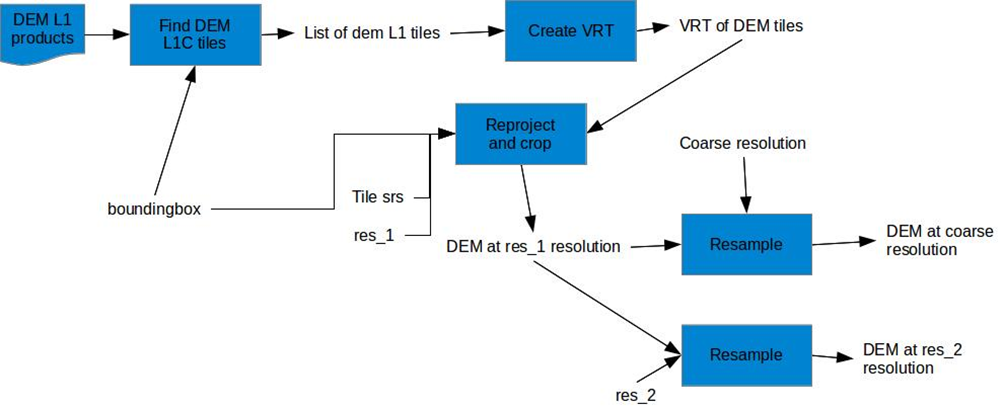


Figure 4‑15 : DEM L2 Computation

###### Create VRT

This sub-component uses the *gdalbuildvrt* command with the following syntax to merge the dem tiles in one vrt file:

More information about the command can be found in [RD.11].

gdalbuildvrt <outputFile.vrt> <input\_dem\_1> <input\_dem\_2>...<input\_dem\_n>

###### Reproject and Crop

This step requires the OTB application OrthoRectification. From the extent and the resolution, extract the size of the output image.

otbcli\_OrthoRectification

-io.in dem.vrt

-io.out dem\_r1.tif

-map utm

-map.utm.zone number\_of\_utm\_zone

-map.utm.northhem False

-outputs.spacingx spx

-outputs.spacingy spy

-outputs.ulx extent\_upper\_left\_corner\_x

-outputs.uly extent\_upper\_left\_corner\_y

-outputs.sizex x\_size

-outputs.sizey y\_size

More information about this application can be found in [RD.5][.](https://www.orfeo-toolbox.org/CookBook/CookBooksu38.html" \l "x59-810004.2.4)

###### Resampling

* This sub-component is described in 4.2.21

##### Compute L2 Water Body

Then, the second submodule consists in computing aspect and slope at the right resolution as shown in Figure 4 -16.

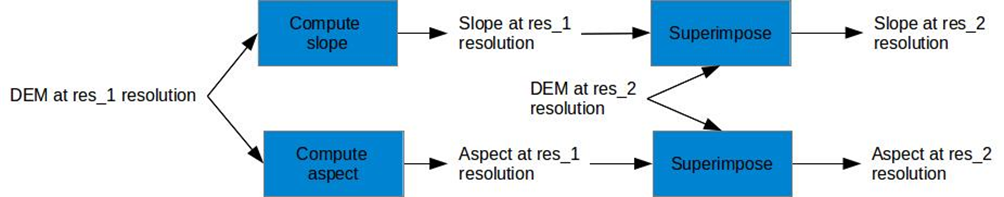


Figure 4‑16 : DEM L2 Water Bodies Computation

###### Computing Aspect and Slope

This submodule uses *gdaldem* to perform aspect and slope.

gdaldem slope dtm\_r1.tif -scale 111120 -compute\_edges slope\_r1.tif

gdaldem aspect dtm\_r1.tif -scale 111120 -compute\_edges aspect\_r1.tif

See [RD.11] for more information.

###### Superimpose

This submodule requires the OTB application Superimpose to resample the image to the right resolution and extent.

otbcli\_Superimpose

-inr DEM\_r2.tif

-inm slope\_r1.tif

-interpolator linear

-lms 40

-out slope\_r2.tif

For more information about the OTB application, see [RD.5]

##### Compute L2 Water Body

* This submodule leans on Figure 4 -17:

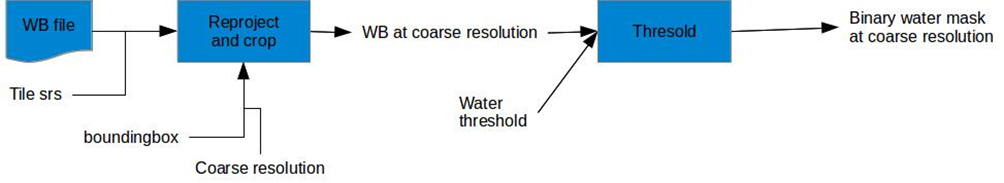


Figure 4‑17 : DEM L2 Water Bodies Computation (2)

###### Threshold

The threshold consists in an OTB BandMath with the following expression:

if(im1b1>water\_threshold, 1,0)

For more information about the OTB application BandMath, see [RD.5]

The reproject and crop function is the same that in the component compute L2 module.

#### Dependencies

The dependences of this module are the following:

* Common library to provide capabilities to read, copy, update a product
* OTB application package
* GDAL tools

All these components should be initialized at the beginning of the module.

#### Interfaces

The main interfaces in term of control flow are:

* The job order which indicates the main information about the processing input and output data, the location of log files and the location of the working directory.

The main interfaces in term of data flow of its module are:

* As input, a L1 DEM product:
  + n L1 DEM tiles
* As output, a L2 DEM product in a format compatible with MACCS:
  + a DEM at the given r1 resolution,
  + a DEM at the given coarse resolution
  + a DEM at resolution r2 for Sentinel 2 products
  + slope at r1 resolution (and r2 for Sentinel 2 products)
  + aspect at r1 resolution (and r2 for Sentinel 2 products)
  + water mask at coarse resolution
* As auxiliary data
  + Not applicable here
* As image processing parameters
  + Resolutions r1, r2 and coarse
  + Input tile coordinate system
  + Input tile extent
  + Water threshold

#### Resources

Not Applicable

The product format of the DEM\_L1 product is based on the CGIA-CSI one for the internal and on the USGS [RD.8].

This component used also some common sub-components from the Common package/library: resample image data.

#### Data

The internal data of this component will be store into the working directory until the end of the module execution.

The current list of internal data is the following:

* VRT file of DEM tiles
* Water body at coarse resolution

### L2A S2 Pre Processing Module

#### Type

The L2A S2 Pre Processing Module is a module which represents the pre-processing step of the L2A processor specific to Sentinel 2. This module is a unique function which is put in a specific module to allow processing by the tile in further module.

#### Purpose

The component consists to split a L1C product which can contain n (≥1) tiles into n L1C products.

Note: a product contains tiles from only one orbit.

#### Function

This module is very simple due to the L1C product format [RD.9]. Indeed the L1C product format is a collection of independent tiles referenced in the product xml. Therefore the function consist to create from a L1C product a new L1C products by copy of all the elements of the considered tile, the datastrip and update the product metadata file (modify the Product\_Organisation/Gralule\_list/Granules item).

#### Subordinates

Not Applicable

#### Dependencies

This module depends only from the Common library/package to provide capabilities to read, copy, update a L1C product.

This component should be initialized at the beginning of the module.

#### Interfaces

The main interfaces in term of control flow are:

* The job order which indicates the main information about the processing input and output data, the location of log files and the location of the working directory.

The main interfaces in term of data flow of its module are:

* As input, a L2A product (S2\_L2A\_PRD):
* As output, a new set of L2A product (S2\_L2A\_PRD) mono tile:
* As auxiliary data
  + Not applicable here
* As image processing parameters
  + Not applicable here

##### executeAsync

Requests the asynchronous execution of the L2A S2 Pre Processing Module by the Resource Manager.

Input:

* Parameters []: the list of processor parameters as described in the control and data flow.

##### stop

Requests the stopping of the L2A S2 Pre Processing Module.

##### inspectState

Requests information about the current execution state of the L2A S2 Pre Processing Module.

Output:

* ProcessorState : the execution information of the processor

#### Resources

Not Applicable

#### References

This module is mainly based on the Sentinel-2 PSD document.

#### Data

Not Applicable

### L2A MACCS Processing Module

#### Type

The L2A MACCS Processing is a main module of the L2A processor. This module is based on the CFI tool MACCS.

#### Purpose

The L2A MACCS Processing consists to generate from [1..n] S2 L1C products or [1..n] L8 L1T a L2A product.

#### Function

The MACCS software has 3 different processing modes:

* Init Mode which allow to generate a L2A at date D from a L2A at date D.
* Nominal mode which allow to generate a L2A at date D from a L1C at data D and a L2A at data D-1
* Backward mode which allow to generate a L2A at date D from a set of L1C acquired between D and D + N

This mode is specified in the job order provided to MACCS.

The L1C product format follows the PDGS format for the S2 and the USGIS one for L8.

The L2A product format follows the MACCS L2A format defined for S2 and L8.

#### Subordinates

The only subordinate is the MACCS CFI.

#### Dependencies

This module depends from the MACCS package.

This component should be installed and initialized at the beginning of the module.

#### Interfaces

The main interfaces in term of control flow are:

* The job order which indicates the main information about the processing input and output data, the location of log files and the location of the working directory.

The main interfaces in term of data flow of its module are:

* As input, one or several L1C product (S2/L8\_L1C\_PRD) and optionally a S2/L8\_L2A\_PRD
* As output, a new L2A product (S2/L8\_L2A\_PRD)
* As technical data
  + DEM\_L2\_PRD
  + MACCS GIPP
  + Configuration Parameters

#### Resources

Not Applicable

#### References

This module is mainly on the User Manual [RD.10] of MACCS.

#### Data

Not Applicable

### L2A L8 Post Processing Module

#### Type

The L2A L8 Post Processing is a module which represents the post-processing step of the L2A processor specific to LandSat 8. This module is a composed of set of subroutines.

#### Purpose

The L2A L8 Post Processing consists to split a L2A L8 product into *n(≥1)* L2A S2-L8 products.

#### Function

This module is composed of three steps:

* Identify the list of corresponding S2 tiles covered by the L8 tile;
* Reproject and crop the L8 tile (image data, mask data) to the different identified S2 tiles;
* Format the new set of products.

The first item will be based on auxiliary data previously computed:

* A list of S2 tiles covered by each possible L8 tile : L8\_S2\_tile\_ID\_list\_vX.Y.xml;
* A list composed of the bounding box of each S2 tiles : S2\_tile\_BB\_list\_vX.Y.xml;
* A table which identified for each L8 band ID the corresponding S2 band IDs: L8band\_S2bands\_vX.Y.xml;
* A table which identified for each S2 band ID the corresponding spatial resolution: S2bands\_res\_vX.Y.xml/

The pseudo-code of this module is the following:

S2\_tile\_ID\_list = Read(L8\_S2\_tile\_ID\_list, L8\_tile\_ID)

For each S2\_tile\_ID in S2\_tile\_ID\_list

S2\_tile\_BB = Read(S2\_tile\_BB\_list, S2\_tile\_ID)

For each b\_L8 in b\_L8\_list

S2\_bands\_list = Read(L8band\_S2bands, b\_L8)

For each b\_S2 in S2\_bands\_list

b\_S2\_res = Read(S2bands\_res,b\_S2)

ρS2L8(b\_S2) = ReprojectAndCrop(ρL8(b\_L8), S2\_tile\_BB, b\_S2\_res)

For each MSK in the available masks in the product

For each res in S2\_res\_list

MSKS2L8(res) = ReprojectAndCrop(MSKS2, S2\_tile\_BB, res1)

If MSKS2L8 is a binary mask

MSKBS2L8(res) = Threshold(MSKS2L8(res))

CreateL2AProduct(reflectances, masks)

#### Subordinates

Based on pseudo-code above, the list of sub-components is the following:

* Read: this sub-component allow to extract from an xml file some information for a specific key
* ReprojectAndCrop: this sub-component is already described in a previous component;
* Threshold: this sub-component is based on the OTB BandMath application with a simple formula:

if(input≥threshold\_value,1,0)

* CreateL2AProduct: this sub-component allow to create a new L2A product from the available image and mask data into a new directory.

#### Dependencies

The dependences of this module are the following:

* The Common library/package to provide capabilities to read or write L2A product (image and mask data and metadata).
* The OTB application package
* The GDAL API (TBC)

This component should be initialized at the beginning of the module.

#### Interfaces

The main interfaces in term of control flow are:

* The job order which indicates the main information about the processing input and output data, the location of log files and the location of the working directory.

The main interfaces in term of data flow of its module are:

* As input, a L8 L2A product (L8\_L2A\_PRD):
  + Its reflectance
  + Its masks
* As output, a new set of S2 L2A product (S2L8\_L2A\_PRD):
* As auxiliary data
  + L8\_S2\_tile\_ID\_list
  + S2\_tile\_BB\_list
  + L8band\_S2bands
  + S2bands\_res
* As image processing parameters
  + Not applicable here

#### Resources

Not Applicable

#### References

This module is mainly based on the MACCS L2A PSD document and the documentation of the OTB application.

#### Data

Not Applicable

### Bidirectional Correction Processor

#### Type

The bidirectional correction is a module which represents an optional pre-processing step of the L3A processor. This module is a collection of executable or internal functions.

#### Purpose

The bidirectional correction consists to correct the effect of the large set of viewing and illumination angles available in a tile and the effect induce by the fact that several part of a tile are viewed by different orbits and so with different angles. The impact of this effect is quite important during the composite synthesis [RD.1].

Currently, due to the lack of angles data grid in Landsat 8, only Sentinel-2 data are corrected.

#### Function

This module is based on the algorithm description done in the Composite ATBD [RD.1]. In Figure 4 -18, we present a global diagram of the different internal steps of the module and the main data elements.

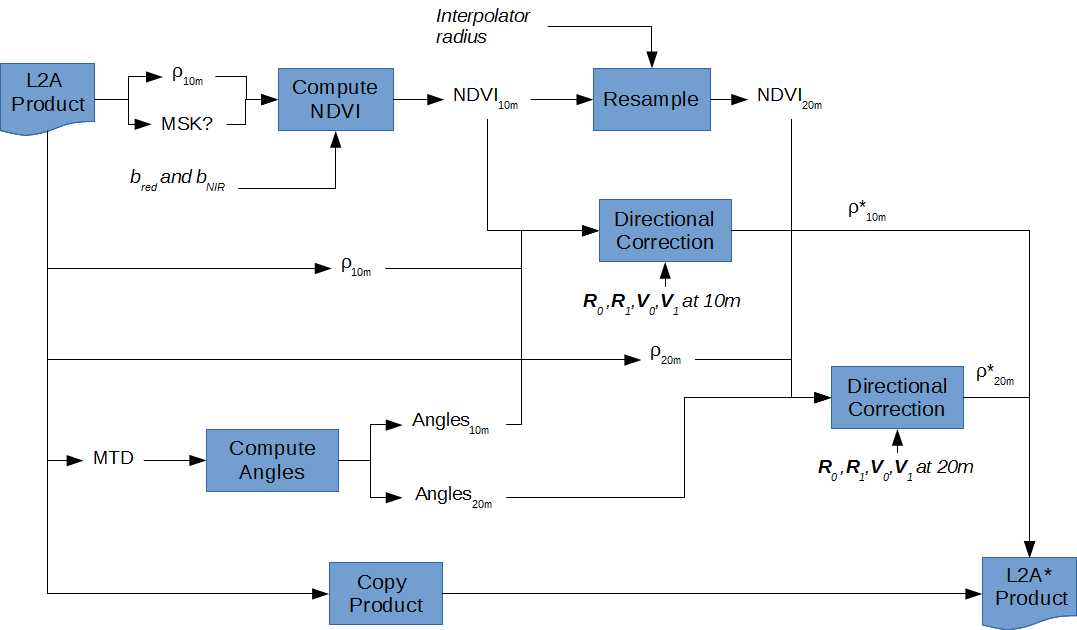


Figure 4‑18 : Bidirectional Correction Processor Diagram

The different processing steps and interface are described below.

#### Subordinates

Based on Figure 4 -18, the list of sub-components is the following:

* Compute NDVI: this sub-component is described in section 4.2.20;
* Resample: this sub-component is described in section 4.2.21 ;
* Compute Angles Grid: this sub-component is below;
* Directional Correction: this sub-component is below;
* Copy Product: this sub-component allow to create a new product by copy of all its elements following the input format into a new directory;
* Update Product: this sub-component allow to update a product with new set of information: here new image data content.

##### Compute Grid Angle

This sub-component extracts the angles from the metadata of the L2A product and generates the viewing and the illumination angles at the two S2 spatial resolutions. This component has already been developed in the frame of the Sentinel-2 MPA project according to the specification of the Sentinel-2 product format. In Figure 4 -19, the main steps of this sub-component are described.

This sub-component uses the *gdal\_merge* command with the following syntax to merge the grid angles from each detector:

gdal\_merge.py

-n <nodata\_val>

-a\_nodata <output\_nodata\_val>

-o <outputFile>

<inputFile1> <inputFile2> <inputFile...>

More information about the command can be found in [RD.11].

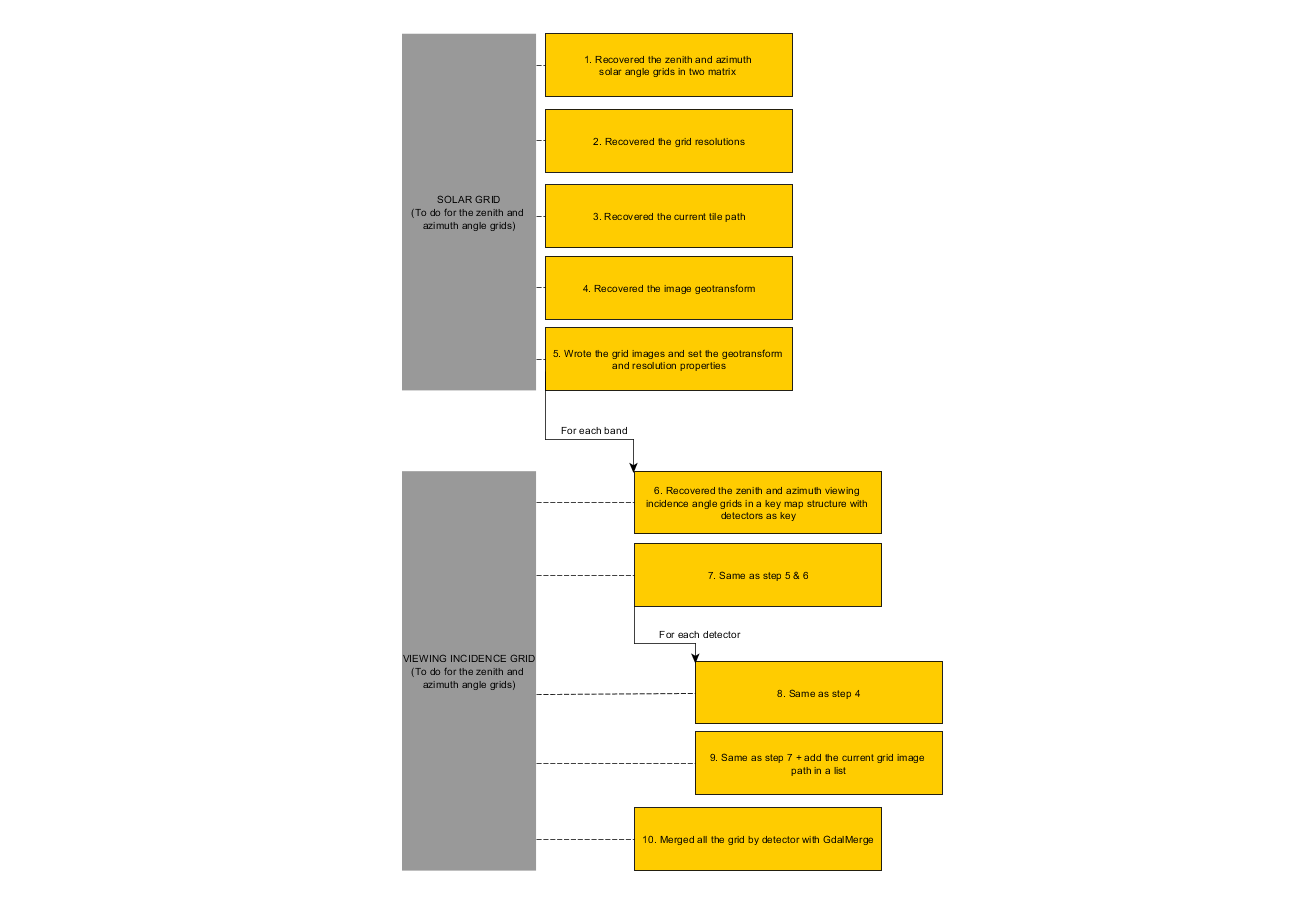


Figure 4‑19 : Compute Grid Angle Sub-Component Steps

##### Directional Correction

This sub-component is described in the Composite ATBD [RD.1]. The pseudo-code available in the ATBD can be defined by a formula compatible with the BandMathX application [RD.5] available in OTB.

#### Dependencies

The dependences of this module are the following:

* Common library to provide capabilities to read, copy, update a product;
* OTB application package;
* GDAL tools.

All these components should be initialized at the beginning of the module.

#### Interfaces

The main interfaces in term of control flow are:

* The job order which indicates the main information about the processing input and output data, the location of log files and the location of the working directory.

The main interfaces in term of data flow of its module are:

* As input, a Sentinel-2 L2A product (S2\_L2A\_PRD):
  + Its image data: the reflectance ρ at the 2 resolutions
  + Its mask data: S2\_L2A\_MSK (TBC)
  + Its metadata: S2\_L2A\_MTD (for the viewing and illumination angles grid)
* As output, a Sentinel-2 L2A product (S2\_L2A\*\_PRD):
  + A new set of reflectance ρ\* at the 2 resolutions
  + The same mask data that the input product
  + The same set of metadata that the input product except the level product which is now L2A\*/c (TBD)
* As auxiliary data
  + Not applicable here
* As image processing parameters
  + Coefficients of the bidirectional model used as described in the Composite ATBD [RD.1]
  + Index of the spectral bands used to compute the NDVI
  + Radius of the interpolator used to resample the NDVI to 20 m.

##### executeAsync

Requests the asynchronous execution of the Bidirectional Correction Processor by the Resource Manager.

Input:

* Parameters []: the list of processor parameters as described in the control and data flow including the next state to be executed.

##### stop

Requests the stopping of the Bidirectional Correction Processor.

##### inspectState

Requests information about the current execution state of the Bidirectional Correction Processor.

Output:

* ProcessorState : the execution information of the processor

#### Resources

Not Applicable

#### References

The algorithm part of this component is described in the Composite ATBD [RD.1].

This component used also some common sub-components from the Common package/library: resample image or mask data, compute a spectral indices (here NDVI).

#### Data

The internal data of this component will be stored into the working directory from the processing area file system until the end of the module execution.

The current list of internal data is the following:

* NDVI at 10m and 20m
* Viewing and illumination angles grid at 10 and 20m
* ρ\* at 10 and 20m

### Iterative Composite Synthesis Processor

#### Type

The Iterative Composite Synthesis Processor is a module which represents the main processing step of the L3A processor. This module is a collection of executable or internal functions.

#### Purpose

The iterative composite synthesis consists to generate an updated L3A product after the production of a new L2A data: after bidirectional correction module for Sentinel-2 data and directly after L2A processing for Landsat 8. This module support to update the L3A product from different sensor based on the Sentinel-2 data.

Due to the fact that the composite period is a parameter of the system (between 30 and 50 days) and that the delivery period is monthly, it should be notice that in some case, for one new L2A product, we should update two L3A product.

#### Function

This module is based on the algorithm description done in the Composite ATBD [RD.1]. In Figure 4 -20, we present a global diagram of the different main steps of the module. The different sub-components describes in the following section.

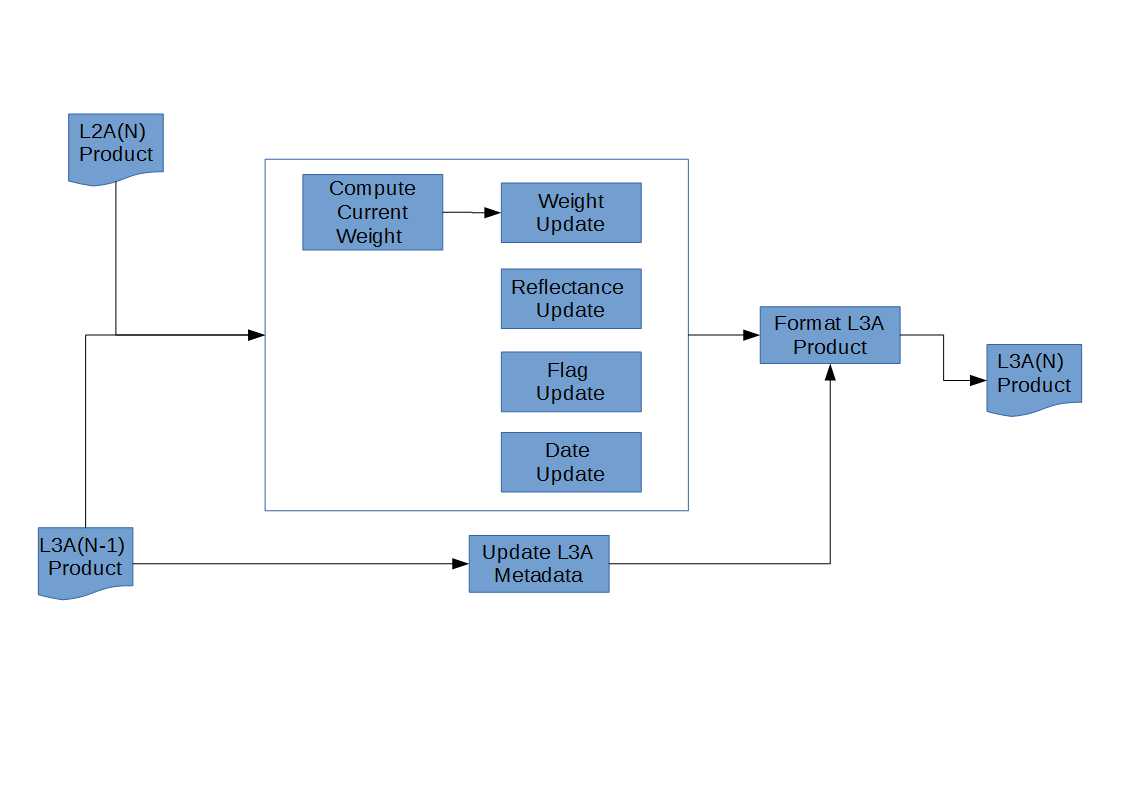


Figure 4‑20 : Iterative Composite Synthesis Processor Functional Diagram

#### Subordinates

Based on the Figure 4 -20 the list of sub-components is the following:

* Compute Current Weight: this sub-component is below
* Weight Update: this sub-component is described below
* Flag Update this sub-component is below
* Date update: this sub-component is below
* Format L3A Product: this sub-component allow to format a new L3A product following the L3A format into a new directory.
* Update L3A Metadata: this sub-component allow to update a product with a new set of metadata information: here insert a new date in the data list.

##### Compute Current Weight

This sub-component compute at the two S2 spatial resolutions, the current weights used in the weighs update and in the reflectance weighted synthesis. The algorithmic part of this sub-component and its internal function are described in the Composite ATBD [RD.1]. We present in Figure 4 -21, the external and internal interface of this sub-component.

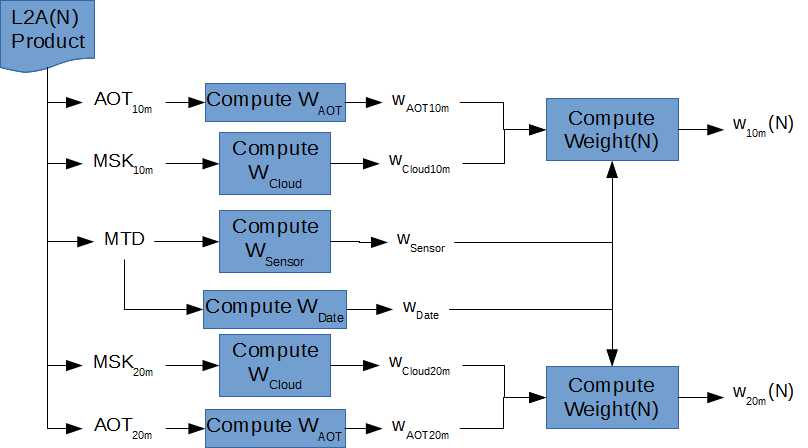


Figure 4‑21 : Compute Current Weight Sub-Component Logical Diagram

Compute Weigh AOT and Compute Weight (N) functions, will use an OTB BandMath application with the formula available in the ATBD [RD.1]. Compute Weight Cloud function, will use a new OTB application based on an existing set of OTB classes. Compute Weight Sensor and Compute Weight Date functions can be written in few python code lines.

##### Weight Update

This sub-component computes at the two S2 spatial resolutions, the updated weights used during the reflectance update step. The algorithmic part of this sub-component is described in the Composite ATBD [RD.1]. The Figure 4 -22 presents the main interfaces of this sub-component. The inputs and the output of this sub-component are image data. Moreover the algorithm used is pixel-based, we propose to re-use the OTB BandMath Application with the corresponding formula.

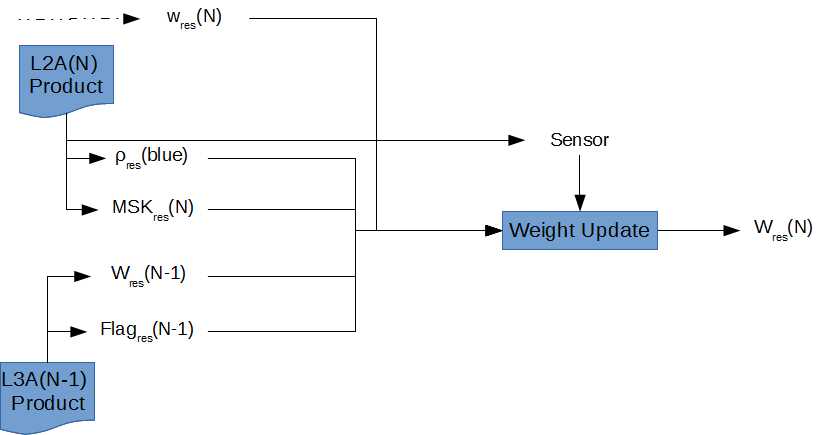


Figure 4‑22 : Weight Update Sub-Component Diagram

##### Flag Update

This sub-component computes at the two S2 spatial resolutions, the temporal status of the pixel. The algorithmic part of this sub-component is described in the Composite ATBD [RD.1]. Figure 4 -23 presents the main interfaces of this sub-component. The inputs and the output of this sub-component are image data. Moreover the algorithm used is pixel-based, so we propose to re-use the OTB BandMath Application with the corresponding formula.

It should be notice that for the 20m spatial resolution, we should resample the current blue reflectance and the mean blue reflectance. To do that we re-use the common resample function of the common library/package.

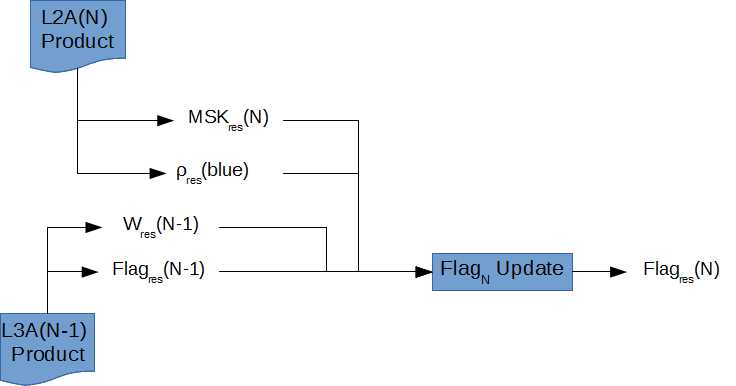


Figure 4‑23 : Flag Update Sub-Component Diagram

##### Date update

This sub-component computes at the two S2 spatial resolutions, the virtual date of generation of the pixel. The algorithmic part of this sub-component is described in the Composite ATBD [RD.1]. Figure 4 -24 presents the main interfaces of this sub-component. The inputs and the output of this sub-component are image data. Moreover the algorithm used is pixel-based, so we propose to re-use the OTB BandMath Application with the corresponding formula.

It should be noticed that for the 20m spatial resolution, we should resample the current blue reflectance and the mean blue reflectance. To do that we re-use the common resample function of the common library/package.

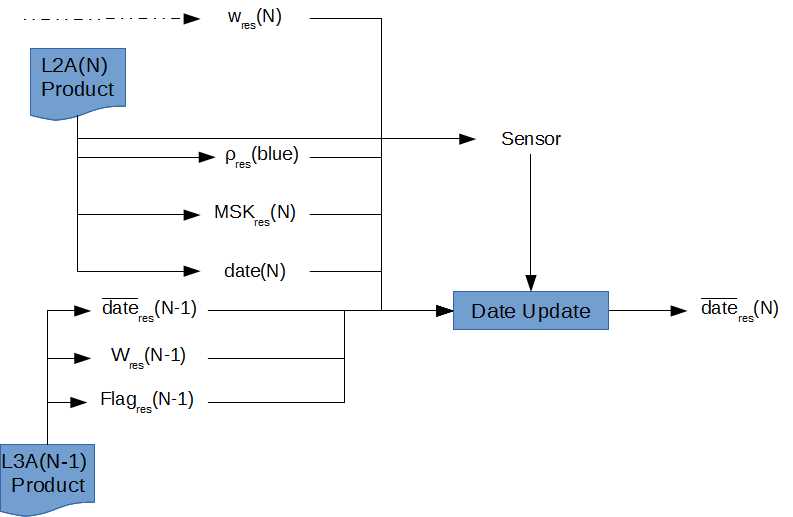
a

Figure 4‑24 : Date Update Sub-Component Diagram

#### Dependencies

The dependences of this module are the following:

* Common library/package to provide capabilities to read, copy, update a product and simple mathematical operation.
* OTB application package
* S2Agri application package (for the new ComputeWeightCloud application)
* GDAL tools

All these components should be initialized at the beginning of the module.

#### Interfaces

The main interfaces in term of control flow are:

* The job order which indicates the main information about the processing input and output data, the location of log files and the location of the working directory.

The main interfaces in term of data flow of its module are:

* As first input, the previous L3A product (S2\_L3A\_PRD):
  + Its image data: the reflectance ρ at the 2 resolutions
  + Its mask data: S2\_L3A\_MSK
* As second input, the current L2A or L2A\*/c (TBC) product (S2\_L2A\_PRD)
  + Its image data: the reflectance ρ at the 2 resolutions
  + Its mask data: S2\_L2A\_MSK
  + Its metadata: S2\_L2A\_MTD (for the sensor type and the acquisition date)
* As output, the new L3A product (S2\_L3A\_PRD):
  + A new set of reflectance ρ at the 2 resolutions
  + A new set of mask data
  + The same set of metadata that the previous product, except for the insertion of a new date in the date list.
* As auxiliary data
  + Not applicable here
* As image processing parameters
  + Radius of the interpolator used to resample the blue reflectance to 20 m.
  + The two sigma values used into Compute Weight Cloud
  + The coarse resolution used into Compute Weight Cloud

##### executeAsync

Requests the asynchronous execution of the Iterative Composite Synthesis Module by the Resource Manager.

Input:

* Parameters []: the list of processor parameters as described in the control and data flow including the next state to be executed.

##### stop

Requests the stopping of the Iterative Composite Synthesis Module.

##### inspectState

Requests information about the current execution state of the Iterative Composite Synthesis Module.

Output:

ProcessorState : the execution information of the processor

#### Resources

Not Applicable

#### References

The algorithm part of this component is described in the Composite ATBD [RD.1]

This component used also some common sub-components from the Common package/library: resample image or mask data.

#### Data

The internal data of this component will be stored into the working directory of the processing area file system until the end of the module execution.

The current list of internal data is the following:

* The current AOT weight map at 10m and 20m
* The current cloud weight map at 10m and 20m
* The current global weight map at 10m and 20m
* The mean weight map at 10m and 20m
* The mean date map at 10m and 20m
* The temporal flag at 10 and 20m
* The update reflectance for the 10m and 20m components
* The blue reflectance at 20m

### Compute LAI

#### Type

The compute LAI module represents the main processing step of the vegetation status processor. This module is a collection of executable or internal functions.

#### Purpose

The compute LAI module provides a biophysical measure that characterizes vegetation for each available pixel.

#### Function

This module is based on the algorithm description done in the Biophysical Product ATBD [RD.2]. In Figure 4 -25, we present a global diagram of the different main steps of the module. The different sub-components are described in the following section.

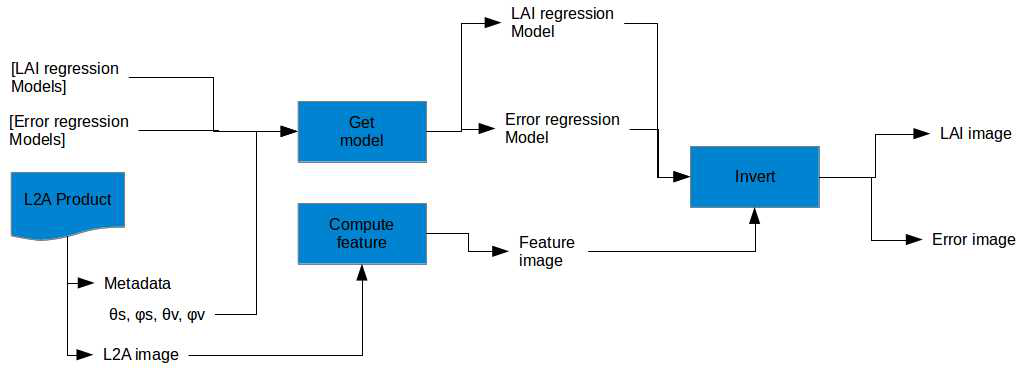


Figure 4‑25 : Compute LAI Module Functional Diagram

#### Subordinates

Based on the Figure 4 -25, the list of sub-components is the following:

* Get model: this sub-component is described below
* Compute feature: this sub-component is described below
* Invert: this sub-component is described in [RD.2]

##### Get Model

This sub-component is described below (Figure 4 -26).

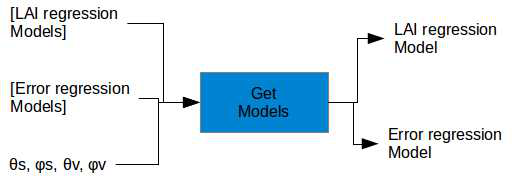


Figure 4‑26 : LAI Get Model Sub-Component Diagram

Get Models looks for the right error and LAI regression model from lists according to angles given as inputs. The selection is made on the filename. As the models are computed with a step of 0.5 degrees [TBC], the module takes the nearest models according to:

* For Sentinel-2 the mean angles per band and detectors for viewing angles and sun angles as available in the Sentinel-2 tile metadata.
* For LandSat8 the global viewing angles and sun angles

##### Compute Feature

This sub-component uses the method described in section 4.2.20.

The requested indices are “ndvi” and “rvi”.

#### Dependencies

The dependence of this module is the following:

* The Common library/package to provide capabilities to read a L2A product (image and mask data and metadata) and write LAI tile product
* OTB application package

This component should be initialized at the beginning of the module.

#### Interfaces

The main interfaces in term of control flow are:

* The job order which indicates the main information about the processing input and output data, the location of log files and the location of the working directory.

The main interfaces in term of data flow of its module are:

* As inputs, a L2A product:
  + Image data
  + θs, φs, θv, φv (singles values for LandSat8 and grid values for Sentinel-2)
  + masks
* As output: a LAI tile product with:
  + A LAI Image
  + A Error Image
* As auxiliary data
  + List of all LAI Regression Models for the considered sensor
  + List of all Error Regression Models for the considered sensor
* As image processing parameters
  + Not applicable

#### Resources

Not Applicable

#### References

The algorithm part of this component is described in the Biophysical Product ATBD [RD.2]

#### Data

The internal data of this component will be stored into the working directory until the end of the module execution.

The current list of internal data is the following:

* NDVI, RVI

### Reprocess LAI

#### Type

The LAI reprocessing represents a refining step of the vegetation status processor. This module is a collection of executable or internal functions.

#### Purpose

The LAI reprocessing module allows to refine the LAI value of a pixel according to set of previous LAI data.

#### Function

This module is based on the algorithm description done in the Biophysical Product ATBD [RD.2]. In Figure 4 -27, we present a global diagram of the different main steps of the module. The different sub-components describes in the following section.



Figure 4‑27 : Reprocess LAI Functional Diagram

#### Subordinates

Based on the Figure 4 -27, On-line retrieval is the sub-component of this module.

This sub-component uses the OTB Application otbcli\_ProfileReprocessing with the *algo* mode set at *local* as it follows:

otbcli\_ProfileReprocessing

-ipf $inputprofile

-opf $outputprofile

-algo local

-algo.local.bwr 2

-algo.local.fwr 0

For more information, see [RD.2 section 2.1.1.1.5].

#### Dependencies

The dependence of this module is the following:

* The Common library/package to provide capabilities to read or write LAI product (image and metadata)
* OTB application package

This component should be initialized at the beginning of the module.

#### Interfaces

The main interfaces in term of control flow are:

* The job order which indicates the main information about the processing input and output data, the location of log files and the location of the working directory.

The main interfaces in term of data flow of its module are:

* As inputs:
  + A time series LAI Images from date t-x to date t to process image from date t
  + A time series Error Images from date t-x to date t to process image from date t
* As output, a reprocessed LAI tile product (mainly update it LAI image content)
* As auxiliary data
  + Not applicable here
* As image processing parameters
  + Not applicable here

#### Resources

Not Applicable

#### References

The algorithm part of this component is described in the Biophysical Product ATBD [RD.2]

#### Data

Not Applicable here.

### NDVI Metrics

#### Type

The NDVI metrics module represents an additional processing step of the vegetation status processor preformed at the end of the season. This module is a collection of executable or internal functions.

#### Purpose

The NDVI metrics module allows extracting metrics from L2A reflectance time series.

#### Function

This module is based on the algorithm description done in the Biophysical Product ATBD [RD.2]. In Figure 4 -28 , we present a global diagram of the different main steps of the module. The different sub-components describes in the following section.

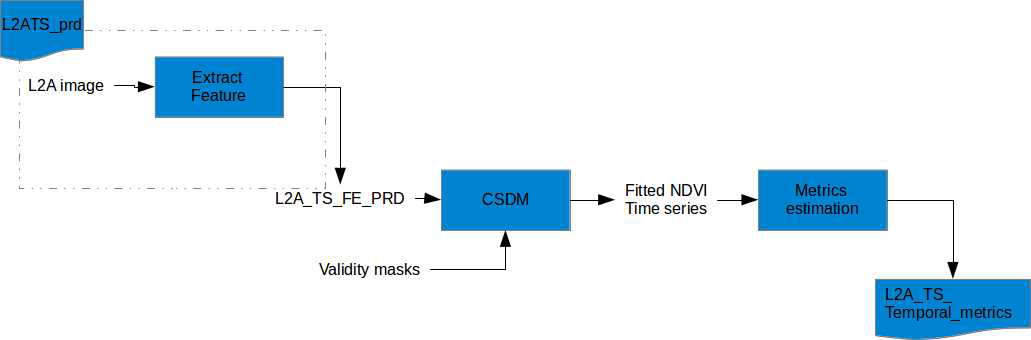


Figure 4‑28 : Functional diagram of the NDVI metrics module

#### Subordinates

Based on the Figure 4 -28, the list of sub-components is the following:

* Extract feature: this sub-component is described bellow
* CSDM: this sub-component is described bellow
* Metrics estimation: this sub-component is described in [RD.2 section 2.2.4.5]

##### Extract Feature

This sub-component use the method described in section 4.2.20.

The requested feature is “ndvi”.

##### CSDM

This sub-component is in two parts.

First, the invalid pixels are removed with the OTB application BandMath with the following expression: (if one of the mask is at 1, set output to 0)

if(im2b1 or im3b1 or im4b1 or ..., 0, im1b1)

Then, it uses the OTB Application otbcli\_ProfileReprocessing with the *algo* mode set at *fit* as it follows:

otbcli\_ProfileReprocessing

-ipf $inputprofile

-opf $outputprofile

-algo fit

For more information, see [RD.2 section 2.2.3].

#### Dependencies

The dependence of this module is the following:

* The Common library/package to provide capabilities to read a L2A product (image and mask data) and write a new element (ndvi metrics map) in a LAI product
* OTB application package

This component should be initialized at the beginning of the module.

#### Interfaces

The main interfaces in term of control flow are:

* The job order which indicates the main information about the processing input and output data, the location of log files and the location of the working directory.

The main interfaces in term of data flow of its module are:

* As inputs, a L2A Time Series
* As output, a L2A TS temporal metrics
* As auxiliary data
  + Not applicable here
* As image processing parameters
  + Not Applicable

#### Resources

Not Applicable

#### References

The algorithm part of this component is described in the Biophysical Product ATBD [RD.2]

This component used also some common sub-components from the Common package/library: compute spectral indices.

#### Data

The internal data of this component will be stored into the working directory until the end of the module execution.

The current list of internal data is the following:

* A L2A\_TS\_FE\_PRD which contains n dates
* n fitted NDVI with n equal to the number of input products

### Samples Preparation

#### Type

The samples preparation is a module which represents a processing step of the supervised classification processing. This module is a collection of executable or internal functions.

#### Purpose

The data preparation subsystem aims at generating the input data for the supervised learning used in the crop type and crop mask generation [RD.3 section 2.2 & 2.2.1].

#### Function

This module is based on the algorithm description done in the Crop Type ATBD [RD.3]. In Figure 4 -29, we present a global diagram of the different main steps of the module. The different sub-components describes in the following section.

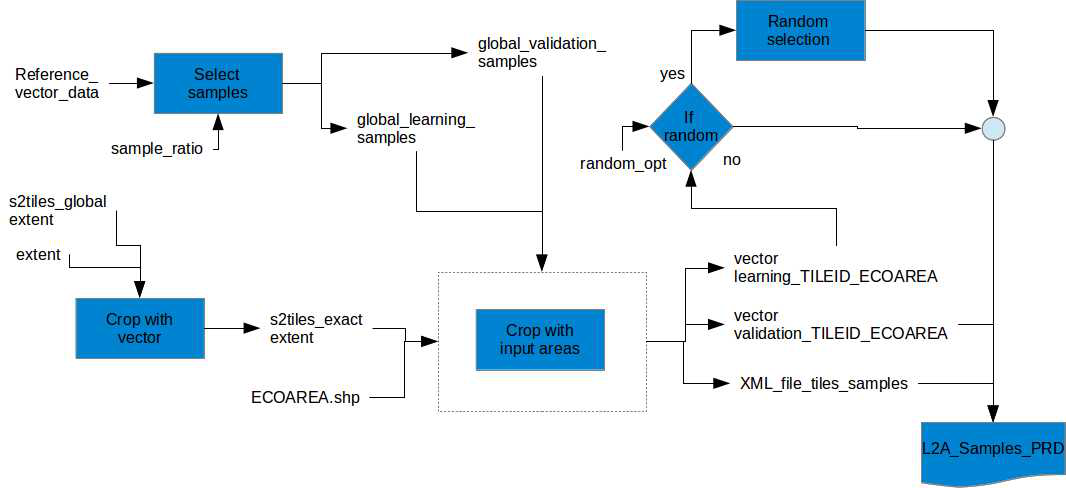


Figure 4‑29 : Samples Preparation Functional Diagram

#### Subordinates

Based on the Figure 4 -29, the list of sub-components is the following:

* Select samples: this sub-component is described in [RD.3]
* Crop vector with vector: this sub-component is described below
* Crop vector with input area: this sub-component is described belowRandom selection: this step is described in [RD.3section 3.3.1.2]

##### Crop Vector with Vector

The gdal tool *ogr2ogr* is used as it follows:

ogr2ogr

-clipsrc aoi.shp s2tiles\_exact\_extent.shp s2tiles\_global\_extent.shp

For more information about the *ogr2ogr*, see [RD.11]

##### Crop Vector with Each Feature

This submodule crop the given sample vector data with all features of the s2tiles vector data.

See the following pseudo-code:

* Initialize a XML file (for each tile id, there are three children “ECOAREA\_ID”, “LEARNING” and “VALIDATION”
* For each tile from s2tiles\_exact\_extent.shp:
  + Create a single shapefile with the tile: tile\_id.shp
  + if a ECOAREA.shp is not provided :
    - test intersection of global\_learning\_sample.shp with the single tile\_id.shp
      * if it is empty:
        + fill XML a Tile\_ID node, with

ECOAREA\_ID=1

learning=0

* + - * else:
        + fill XML a Tile\_ID node, with

ECOAREA\_ID=1

learning=1

* + Crop global\_learning\_sample.shp with the single tile\_id.shp

The previous method *ogr2ogr* can be used as follows:

ogr2ogr

-clipsrc tile\_id.shp validation\_sample\_tileid.shp validation\_sample.shp$

* + - test intersection of global\_training\_sample.shp with the single tile\_id.shp
      * if it is empty:
        + fill XML a Tile\_ID node, with

ECOAREA\_ID=1

training=0

* + - * else:
        + fill XML a Tile\_ID node, with

ECOAREA\_ID=1

training=1

* + Crop global\_validation\_sample.shp with the single tile\_id.shp
* If an input ECOAREA.shp is provided:
  + For each area in ECOAREA.shp:
    - Create a single shapefile with the area: area\_id.shp
    - Test intersection of the training sample with the single area\_id.shp
      * if it is empty:
        + fill XML a Tile\_ID node, with

ECOAREA\_ID=1

learning=0

* + - * else:
        + fill XML a Tile\_ID node, with

ECOAREA\_ID=1

learning=1

* + - Crop the training sample with the single area\_id.shp

The previous method *ogr2ogr* can be used as follows:

ogr2ogr

-clipsrc area\_id.shp validation\_sample\_tileid.shp validation\_sample.shp

* + - Test intersection of the validation sample with the single area\_id.shp
      * if it is empty:
        + fill XML a Tile\_ID node, with

ECOAREA\_ID=1

validation=0

* + - * else:
        + fill XML a Tile\_ID node, with

ECOAREA\_ID=1

validation=1

* + - Crop the validation sample with the single area\_id.shp

#### Dependencies

The dependences of this module are the following:

* The Common library/package to provide capabilities to read or write L2A product (image and mask data and metadata)
* GDAL tools

All these components should be initialized at the beginning of the module.

#### Interfaces

The main interfaces in term of control flow are:

* The job order which indicates the main information about the processing input and output data, the location of log files and the location of the working directory.

The main interfaces in term of data flow of its module are:

* As input:
  + Reference\_vector\_data.shp
  + S2tiles\_AOI.shp
  + AOI.shp
  + ECO\_MODE {1,2,3}
  + ECO\_AREA.shp (optional)
* As output, a L2A\_Samples\_PRD:
  + A list of learning\_TILEID\_ECOAREA.shp
  + A list of training\_TILEID\_ECOAREA.shp
  + XML file describing all samples for all tiles\_id/eco\_area
* As auxiliary data
  + Not applicable here
* As image processing parameters
  + Sample\_ratio
  + Random\_opt

#### Resources

Not Applicable

#### References

The algorithm part of this component is described in the Crop Type ATBD [RD.3]

#### Data

The internal data of this component will be stored into the working directory until the end of the module execution.

The current list of internal data is the following:

* s2\_tile\_extact\_extent.shp
* global\_learning\_samples.shp
* global\_training\_samples.shp
* validation\_tileid.shp
* training\_tileid.shp

### Compute temporal features module

#### Type

The feature extraction is a module which represents a processing step of the crop mask processor. This module is a collection of executable or internal functions.

#### Purpose

This module is used in the Crop Mask processor. It allows computing several temporal and statistics features.

#### Function

This module is based on the algorithm description done in the Crop Mask ATBD [RD.4]. In Figure 4 -30, we present a global diagram of the different main steps of the module. The different sub-components describes in the following section.

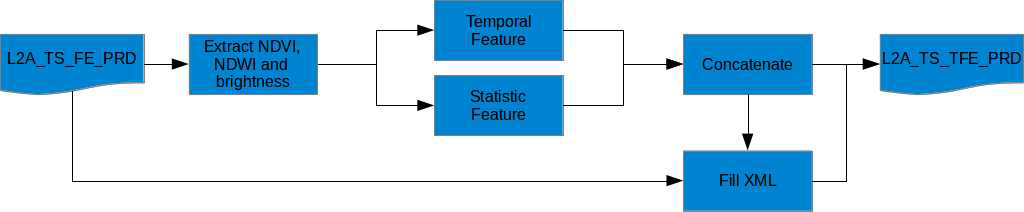


Figure 4‑30 : Temporal Feature Extraction Functional Diagram

#### Subordinates

Based on Figure 4 -30, the list of sub-components is the following:

* Extract NDVI: this sub-component is described below
* Temporal feature: this sub-component is described in [RD.4 section 3.2.1]
* Statistic feature: this sub-component is described in [RD.4 section 3.2.2]
* Concatenate: this sub-component is described in below.
* Fill XML

##### Extract feature component

As a L2A\_TS\_FE\_PRD contains a concatenation of all requested indices, it does not contain exclusively NDVI, NDWI nor brightness indices. L2A\_TS\_FE\_PRD has a descriptive XML file.

* For each feature in list of indices:
  + Read the list of indices and the number of indices and the number of dates from the XML file
  + Let X, the feature in the list of indices, Y the number of indices and Z the number of dates:
    - Get all bands corresponding to : X+mY for m in [0, Z] from data in L2A\_TS\_FE\_PRD

##### Concatenate images

This sub-component uses the OTB application *ConcatenateImages*.

otbcli\_ConcatenateImages -il temporal\_feature.tif statistic\_feaure.tif -out feature.tif.

See RD.5 for more information.

##### Fill XML file

This file has to described the output L2\_TS\_TFE\_PRD:

* Number of dates: length of the L2A\_TS\_TFE\_PRD
* Description of output bands

#### Dependencies

The dependence of this module is the following:

* Common library/package to provide capabilities to read, copy, update a product and simple mathematical operation.
* OTB application package

This component should be initialized at the beginning of the module.

#### Interfaces

The main interfaces in term of control flow are:

* The job order which indicates the main information about the processing input and output data, the location of log files and the location of the working directory.

The main interfaces in term of data flow of its module are:

* As input, a L2A\_TS\_FE\_PRD
  + A data containing X bands, X = nb\_indice\*nb\_dates
  + A XML file describing the previous data file
* As output, a L2A\_TS\_TFE\_PRD
  + A data containing 27 bands
  + A XML file describing the previous data file
* As auxiliary data
  + Not applicable here
* As image processing parameters
  + Not applicable here

#### Resources

Not Applicable

#### References

The algorithm part of this component is described in the Crop Mask ATBD [RD.4]

This component used also some common sub-components from the Common package/library: resample image or mask data.

#### Data

The internal data of this component will be stored into the working directory until the end of the module execution.

The current list of internal data is the following:

* temporal raster feature
* statistic raster feature

### Learning

#### Type

The learning module represents a processing step of common tools of the classification process. This module is a collection of executable or internal functions.

#### Purpose

The learning module allows to get a training model which will be used for the classification step. It is used to generate supervised classification map in the crop type and crop mask processors. The module is used with different inputs for the different cases.

#### Function

This module is based on the algorithm description done in the Crop Type ATBD [RD.3]. In Figure 4 -31, we present a global diagram of the different main steps of the module. The different sub-components describes in the following section.

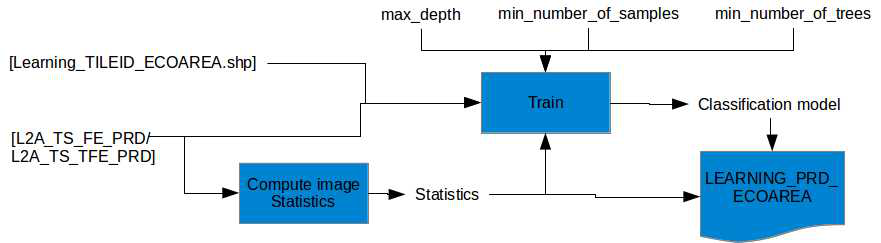


Figure 4‑31 : Learning Module Functional Diagram

It should be noted that in case of L2A\_TS\_TFE\_PRD as input, the eco\_area are not considered.

#### Subordinates

Based on Figure 4 -31, the list of sub-components is the following:

* Computing statistics: this sub-component is below
* Training: this sub-component is below

##### Compute Statistics

This sub component uses the OTB application ComputeImagesStatistics as follows:

otbcli\_ComputeImagesStatistics

-il features.vrt

-out stats.xml

For more information about the OTB application ComputeImagesStatistics, see RD.5

##### Training

This sub-component compute a classification model.

The OTB application TrainImagesClassifier is used as it follows:

otbcli\_TrainImageClassifier

-io.il L2ATS\_FE\_tile01.tif L2ATS\_FE\_tile02.tif L2ATS\_FE\_tile03.tif …

L2ATS\_FE\_tilem.tif

-io.vd learning\_tile01.tif learning\_tile02.tif

learning\_tile03.tif … learning\_tilem.tif

-io.imstat stats.xml

-io.out rf.model

-classifier.rf.max max\_depth

-classifier.rf.min min\_number\_of\_samples

-classifier.rf.nbtrees max\_nb\_trees

For more information about the OTB application TrainImagesClassifier, see RD.5

#### Dependencies

The dependence of this module is the following:

* The Common library/package to provide capabilities to read or write L2A product (image and mask data and metadata)
* OTB application package

This component should be initialized at the beginning of the module.

#### Interfaces

The main interfaces in term of control flow are:

* The job order which indicates the main information about the processing input and output data, the location of log files and the location of the working directory.

The main interfaces in term of data flow of its module are:

* As first input, the data to classify:
  + A list of L2A\_TS\_FE\_PRD or L2A\_TS\_TFE\_PRD
* As second input, the training samples
  + A list of Learning\_TILEID\_ECOAREA.shp
* As output, a LEARNING\_PRD\_ECOAREA
  + A list of statistics files: statistics\_TILEID\_ECOAREA.xml
  + A learning model: Learning\_ECOAREA.model
* As auxiliary data
  + Not applicable here
* As image processing parameters
  + tree\_max\_depth for RF training
  + minimun\_number\_of\_samples for RF training
  + minimun\_number\_of\_trees for RF training

#### Resources

Not Applicable

#### References

The algorithm part of this component is described in the ATBD [RD.5]

This component used also some common sub-components from the Common package/library: resample image or mask data.

#### Data

Not Applicable

### Classification and Validation Module

#### Type

The classification and validation module represents a processing step of the common tools of the classification process. This module is a collection of executable or internal functions.

#### Purpose

The classification allows categorizing pixels of an image according to a given known model. The validation allows checking the relevance of the classification according to validation samples. It is used in crop type and in crop mask generation, only the inputs are different for the two processors and their different strategies.

#### Function

This module is based on the algorithm description done in the Crop type ATBD [RD.3]. In Figure 4 -32, we present a global diagram of the different main steps of the module. The different sub-components describes in the following section.

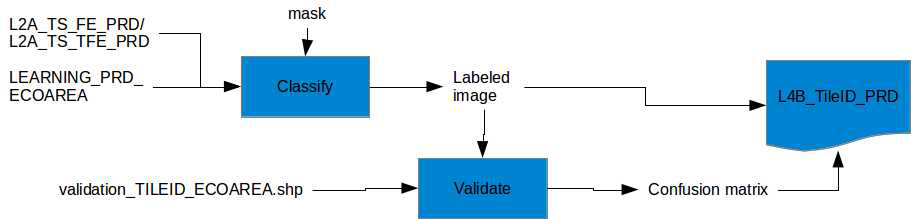


Figure 4‑32 : Classification and Validation Module Functional Diagram

#### Subordinates

Based on Figure 4 -32, the list of sub-components is the following:

* Classify: this sub-component is below
* Validate: this sub-component is described below

##### Classify

This sub-component uses the OTB application *ImageClassifier*:

otbcli\_ImageClassifier

-in L2A\_FE\_TileID.tif

-mask crop\_mask.tif

-imstat stats.xml

–model rf.model

-out crop\_type\_image.tif

See RD.5 for more information.

If as inputs the module has a L2A\_TS\_TFE\_PRD, the mask is empty and not used.

##### Validate

If validation samples are available in the current tile.

This sub-component uses the OTB application *ComputeConfusionMatrix*:

otbcli\_ComputeConfusionMatrix

-in crop\_type\_image.tif

-ref vector

-ref.vector.in validation\_vectors.shp

-ref.vector.field Class

-out crop\_type\_confusion\_matrix.csv

See RD.5 for more information.

#### Dependencies

The dependence of this module is the following:

* The Common library/package to provide capabilities to read or write L2A product (image and mask data and metadata)
* OTB application package

This component should be initialized at the beginning of the module.

#### Interfaces

The main interfaces in term of control flow are:

* The job order which indicates the main information about the processing input and output data, the location of log files and the location of the working directory.

The main interfaces in term of data flow of its module are:

* As inputs:
  + A L2A\_TS\_FE\_PRD or a L2A\_TS\_TFE\_PRD
  + Statistic file of FE\_PRD
  + RF model from training step
  + Validation vector samples
  + Crop mask (optional)
* As output, the new L4A product (S2\_L3A\_PRD):
  + A crop type map
  + A confusion matrix
* As auxiliary data
  + Not applicable here
* As image processing parameters: not applicable here

#### Resources

Not Applicable

#### References

The algorithm part of this component is described in the ATBD [RD.5]

This component used also some common functions from the Common package/library: read or write specific product format.

#### Data

Not Applicable here.

### LUT Generation

#### Type

The LUT generation represents a processing step of the crop type processor. This module is a collection of executable or internal functions.

#### Purpose

The LUT generation allows having a rule to color pixels. The labeled image from classification is colored with this LUT.

#### Function

This module helps to produce a product compatible with the product format of the crop type. In Figure 4 -33, we present a global diagram of the different main steps of the module. The different sub-components describes in the following section.

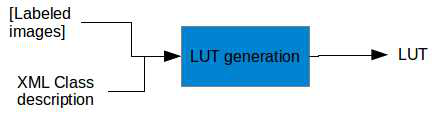


Figure 4‑33 : LUT Functional Diagram

#### Subordinates

Based on Figure 4 -33, the list of sub-components is the following:

* LUT generation: this sub-component is below

##### LUT generation

This subcomponent is used with the following pseudo code:

* For each labeled image:
* For each label of the classification image :
  + Read the color and the label text in the XML
  + Add a new line to the LUT as follows :
    - number\_label red green blue (ex : 1 0 255 45)

#### Dependencies

The dependence of this module is the following:

* The Common library/package to provide capabilities to read or write L2A product (image and mask data and metadata)

This component should be initialized at the beginning of the module.

#### Interfaces

The main interfaces in term of control flow are:

* The job order which indicates the main information about the processing input and output data, the location of log files and the location of the working directory.

The main interfaces in term of data flow of its module are:

* As inputs:
  + List of the classification labelled images
  + XML file with the description of the input chain samples
* As output:
  + A Color Table
* As auxiliary data
  + Not applicable here
* As image processing parameters
  + Not applicable here

#### Resources

Not Applicable

#### References

This component used also some common sub-components from the Common package/library.

#### Data

Not applicable here

### Time Series Feature Extraction Module

#### Type

The feature extraction is a module which represents a processing step commonly used in different Sen2-Agri processors. This module is a collection of executable or internal functions.

#### Purpose

This module is used in the following modules: NDVI metrics, Compute LAI and as pre-processing step of the Crop Type and the Crop Mask processors. It allows to compute several radiometric indices for an input time series.

#### Function

This module is based on the following algorithm:

* For each date from the time series:
  + Resample the bands used in the requested radiometric indices to 10m if necessary.
  + Run OTB application Radiometric Indices with the input indices list and the right band ID.
  + If the input option “add\_reflectance” is equal to True:
    - Resample all the 20m reflectance bands to 10m
    - Concatenate the indices with the input reflectances and the resampled one.
* Concatenate all dates in one file.

This algorithm is illustrated in Figure 4 -34.

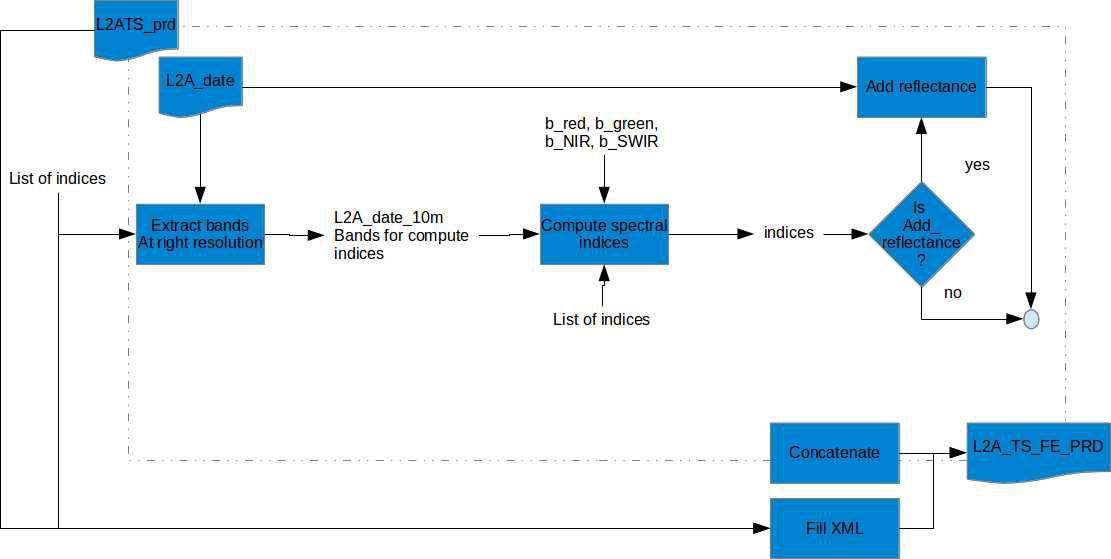


Figure 4‑34: Time Series Feature Extraction Module Functional Diagram

#### Subordinates

Based on the previous algorithm, the list of sub-components is the following:

* Extract bands at right resolution: this sub-component is described below
* Compute spectral indices: this sub-component is below
* Concatenate images: this sub-component is described in a previous module
* Add\_reflectance: this sub-component is described in section 4.2.21
* Fill XML: this sub-component is described below

##### Extract bands at right resolution

This sub-component is described in Figure 4 -35.

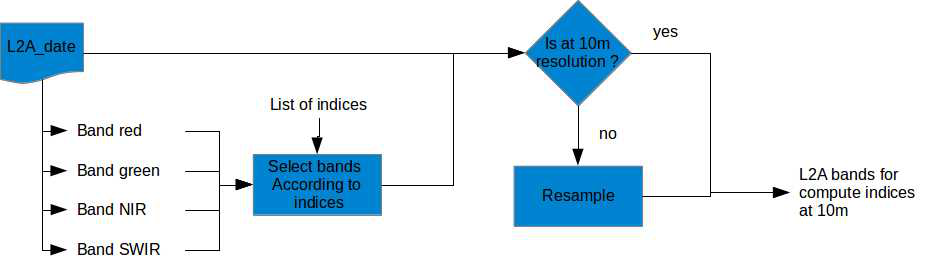


Figure 4‑35: Extract bands at the right resolution functional diagram.

##### Compute Spectral Indices

This module runs the OTB application Radiometric Indices as it follows:

otbcli\_RadiometricIndices

-in tile.tif

-out indices.tif

-channels.blue 1

-channels.green 2

-channels.red 3

-channels.nir 4

-channels.mir 5

-list ndvi ndti

See RD.5, for more information.

##### Concatenate Images

This sub-component uses the OTB applicantion *ConcatenateImages*.

otbcli\_ConcatenateImages

-il ndvi.tif ndwi.tif brightness.tif

-out feature.tif.

See RD.5 for more information.

##### Resample

This sub-component is described in section 4.2.21

##### Add reflectance

Add reflectance is described by the following pseudo code:

* Resample all the 20m reflectance bands to 10m
* Concatenate the indices with the input reflectances and the resampled one.

##### Fill XML file

This file describes the following points:

* Number of dates: length of the L2ATS\_PRD
* Number of indices: length of list\_of\_indices
* Name of indices: list of indices

#### Dependencies

The dependence of this module is the following:

* Common library/package to provide capabilities to read, copy, update the L2A product and create the feature time series internal product.
* OTB application package

This component should be initialized at the beginning of the module.

#### Interfaces

The main interfaces in term of control flow are:

* The job order which indicates the main information about the processing input and output data, the location of log files and the location of the working directory.

The main interfaces in term of data flow of its module are:

* As input, a L2A \_TS\_PRD
* As output, a L2A\_TS\_FE\_PRD:
  + A raster, concatenation of all indices of all dates (and reflectance if requested)
  + A XML file describing computed indices and the number of dates
* As auxiliary data
  + Not applicable here
* As image processing parameters
  + Bands number for red, green, NIR, SWIR according to the sensor
  + List of reflectances
  + Add\_reflectance option

#### References

Not Applicable

#### Data

The internal data of this component will be stored into the working directory until the end of the module execution.

The current list of internal data is the following:

* L2A right bands at 10m resolution
* L2A date spectral features
* L2A date indices concatenated with L2A tile at 10m resolution

### Resampling Module

#### Type

The resampling module is a module which represents a processing step commonly used in different Sen2-Agri processors. This module is a collection of executable or internal functions.

#### Purpose

This module is used a several component or processors to resample data (image or mask) at the desired resolution. This is mainly related to the fact that the Sentinel-2 L2A reflectances used in the system are available at two resolutions 10m and 20m. To keep the higher spatial resolution as much as possible, the resample step is mandatory.

#### Function

This module is based on Figure 4 -36.

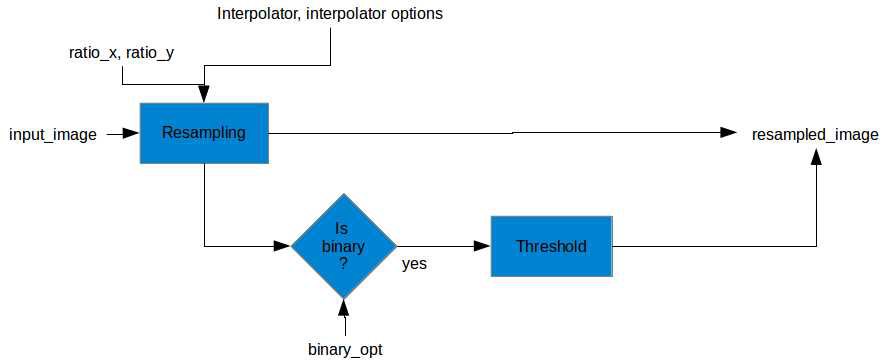


Figure 4‑36 : Resampling Module Functional Diagram

#### Subordinates

Based on the previous algorithm, the list of sub-components is the following:

* Resampling: this sub-component is described below
* Threshold: this sub-component is described below

##### Resampling

This module runs the OTB application Radiometric Indices as it follows:

otbcli\_RigidTransformResample

-in tile.tif

-out tile\_resampled.tif

-transform.type id

-transform.type.id.scalex ratio\_x

-transform.type.id.scaley ratio\_y

-interpolator user\_interpolator

See RD.5, for more information.

##### Threshold

The threshold consists in an OTB BandMath with the following expression:

if(im1b1>bin\_threshold,1,0).

For more information about the OTB application BandMath, see RD.5.

#### Dependencies

The dependence of this module is the following:

* Common library/package to provide capabilities to read the image data from a L2A product.
* OTB application package

These components should be initialized at the beginning of the module.

#### Interfaces

The main interfaces in term of data flow of its module are:

* As input, a raster image
* As output, a raster image resampled
* As auxiliary data
  + Not applicable here
* As image processing parameters
  + ratio\_x, ratio\_y
  + interpolator, interpolator\_options
  + binary\_opt

#### Resources

Not Applicable

#### References

The reference of this module comes from the OTB documentation [RD.5]

#### Data

The internal data of this component will be stored into the working directory until the end of the module execution.

The current list of internal data is the following:

* In case of binary masks, a resampled non-binary image

### CropMask Trimming Data Preparation module

#### Type

The CropMask\_Trimming\_Data\_Preparation represents a processing step of the crop mask processor in the case of we don’t have input field data. This module is a collection of executable or internal functions.

#### Purpose

This module extracts from the input time series a specific subset of data for the trimming operation. This subset of data is related to some temporal features based on the reflectances and the ndvi. This module covers different sections of the Crop Mask ATBD [RD.4 section 4.1 and 4.2].

#### Function

This module is based on the section 4.1 and 4.2 of the ATBD about Crop Mask [RD.4]. The following pseudo code reuses the steps described in the document.

* For each date in the time series:
  + Extract the selected reflectances (and resample to 10m if necessary) at current time and concatenate with previous dates.
  + Compute NDVI at current time and concatenates with the previous dates.
* TemporalSmoothing of the new reflectances time series
* TemporalSmoothing of the NDVI time series
* ComputeReferenceDate according to the smooth reflectances and smooth NDVI time series
* SelectReflectanceFromDate from the smooth reflectances and the reference date

#### Subordinates

The sub-components of this module are:

* PrepareTimeSeries which allow to extract selected reflectances and compute NDVI iteratively. This sub-component is trivial with OTB Applications Extract, BandMath and Concatenate
* TemporalSmoothing which is described in the section 4.1 of the ATBD Crop Mask document
* ComputeReferenceDate which is described in the section 4.2.1 of the ATBD Crop Mask
* SelectReflectanceFromDate which is described in the section 4.2.2 of the ATBD Crop Mask

#### Dependencies

The dependences of this module are the following:

* Common library to provide capabilities to read a L2A product and extract reflectance;
* OTB application package;
* S2Agri application package.

All these components should be initialized at the beginning of the module.

#### Interfaces

The main interfaces in term of control flow are:

* The job order which indicates the main information about the processing input and output data, the location of log files and the location of the working directory.

The main interfaces in term of data flow of its module are:

* As input, a L2A time resampling series product (L2A\_TS\_PRD):
* As output, a set of 5 raster which contains the reflectances at the reference date (L4A\_ref\_tile\_PRD)
* As auxiliary data
  + Not applicable here
* As image processing parameters
  + List of reflectances to use.

#### Resources

Not Applicable here

#### References

The algorithm part of this component is described in the Crop Mask ATBD [RD.4]

#### Data

The internal data of this component will be stored into the working directory from the processing area file system until the end of the module execution.

The current list of internal data is the following:

* Smooth reflectance values
* Smooth and raw NDVI times series
* Reference date for each features

### Reference Map Preparation module

#### Type

The Reference map preparation represents a processing step of the crop mask processor in the case of we don’t have input field data. This module is a collection of executable or internal functions.

#### Purpose

This module prepares for the trimming operation done in the crop mask processing, the list of tiled reference map. This module is done at the beginning of the system when it is known the user AOI and the available reference map. If the user did not provide a reference map, we used the auxiliary references maps which are already tiled, reprojected and eroded.

#### Function

This module is described in the section 4.3 of the ATBD about Crop Mask. We extend the mechanism with the following pseudo code:

* If the user provide a reference map:
  + For each tile in the S2\_AOI\_tile\_list
    - For each classes in the user reference map
      * Reproject and crop the user reference map to the tile extend and projection at 10 m
      * Filter the non-integer values with OTB BandMath application
      * Erode the class image with the OTB Morpho Math application
      * Concatenate with previous classes
    - Write the corresponding tiled\_reference\_map
* Else
  + For each tile in the S2\_AOI\_tile\_list
    - Retrieve the tiled\_reference\_map from the auxiliary data repository
    - Copy (with symbolic link) the corresponding tiled\_reference\_map as output

#### Subordinates

All the sub-component of the pseudo code presented above is already in great majority based on OTB applications. The reproject and crop sub-component is already presented in a previous component.

#### Dependencies

The dependences of this module are the following:

* Common library to provide capabilities to copy data with symbolic link
* OTB application package;
* OGR API to retrieve bounding box from S2\_tiles\_list.

All these components should be initialized at the beginning of the module.

#### Interfaces

The main interfaces in term of control flow are:

* The job order which indicates the main information about the processing input and output data, the location of log files and the location of the working directory.

The main interfaces in term of data flow of its module are:

* As input, the list of tiles covered by the AOI and optionally an user reference map
* As output, a list of tiled reference map
* As auxiliary data
  + System reference map directory
* As image processing parameters
  + Not Applicable here.

#### Resources

Not Applicable

#### References

The algorithm part of this component is described in the Crop Mask ATBD [RD.4 section 4.3]. The OTB applications used in the pseudo-code are available in the OTB CookBook [RD.5].

#### Data

The internal data of this component will be stored into the working directory from the processing area file system until the end of the module execution.

The internal data are the intermediate outputs of the OTB applications.

### Trimming module

#### Type

The trimming represents a processing step of the crop mask processor in the case of we don’t have input field data. This module is a collection of executable or internal functions.

#### Purpose

This module extracts the samples used in the learning step of the Crop Mask processor from a tiled reference map. This module avoids requesting field data from the users for the crop mask.

#### Function

This module is described in the section 4.4 of the ATBD about Crop Mask. The pseudo-code described in this section will be implemented as an OTB application to increase performance. Moreover the trimming process will be done by tiles in order to get a common interface with the learning module.

#### Subordinates

Not Applicable here

#### Dependencies

The dependences of this module are the following:

* Common library to provide capabilities to read a product or create a vector data;
* S2Agri application package

All these components should be initialized at the beginning of the module.

#### Interfaces

The main interfaces in term of control flow are:

* The job order which indicates the main information about the processing input and output data, the location of log files and the location of the working directory.

The main interfaces in term of data flow of its module are:

* As first input, a L4A\_ref\_tile\_PRD
* A secondary input, a tiled\_reference\_map
* As output, a learning\_TILEID vector data
* As auxiliary data
  + Not applicable here
* As image processing parameters:
  + List of classes to trim into the reference map
  + Confidence interval to exclude outlier

#### Resources

Not Applicable

#### References

The algorithm part of this component is described in the Crop Mask ATBD [RD.4].

#### Data

Currently, no internal data of this component are identified.

### Temporal resampling module

#### Type

The temporal resampling is a module which represents a processing step done before time series processing as Crop type and Crop mask. This module is a collection of executable or internal functions.

#### Purpose

This temporal resampling produces a reflectance image time series which is gapfilled with respect to missing data [RD.3 section 2.2.2].

#### Function

This module is based on Figure 4 -37.

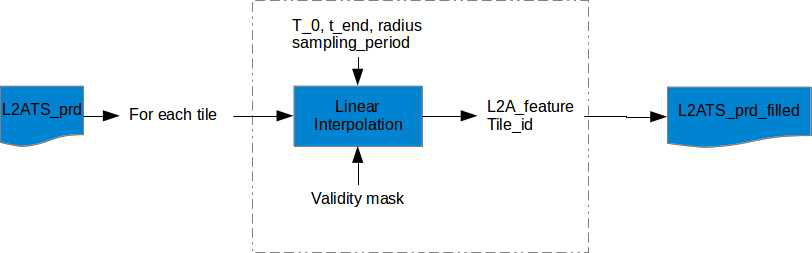


Figure 4‑37 : Temporal resampling functional module

#### Subordinates

Based on the previous algorithm, unique sub-components is Linear interpolation one. This sub-component is described in [RD.3 section 2.2.2].

#### Dependencies

The dependence of this module is the following:

* Common library/package to provide capabilities to read, image data and mask data from a product and to create a new product type.
* OTB application package

This component should be initialized at the beginning of the module.

#### Interfaces

The main interfaces in term of control flow are:

* The job order which indicates the main information about the processing input and output data, the location of log files and the location of the working directory.

The main interfaces in term of data flow of its module are:

* As inputs:
  + a L2A\_TS\_PRD
  + n validity masks for n dates
* As output, a L2A\_TS\_PRD\_filled
* As auxiliary data: not applicable here
* As image processing parameters:
  + t\_0, t\_end
  + sampling\_period
  + radius of the temporal window

#### Resources

Not Applicable

#### References

The algorithm part of this component is described in the Crop Type ATBD [RD.3]

#### Data

Not applicable here

### Crop mask post filtering module

#### Type

The temporal resampling is a post-processing step done optionally after crop mask generation at tile level. This module is a collection of executable or internal functions.

#### Purpose

The Crop mask post filtering module filters spatially the pixel based crop mask to take into account the field object of the scene. Moreover with this module we can define a minimum object size through the segmentation.

#### Function

This post processing step is described in the Crop Mask ATBD [RD.4 section 5]. This module takes as input a NDVI times series and the original pixel based crop mask to generate as output a new object based crop mask at tile level.

#### Subordinates

According to the Crop Mask ATBD, the module is divided in three steps:

* A dimension reduction through a Principal Component Analysis algorithm
* A Mean Shift segmentation
* A object based majority voting

The first two items are based on existing OTB code available in the library and the last one it is quite easy to develop in OTB. Therefore, we will develop this filter and create a new OTB based applications for this post processing step.

#### Dependencies

The dependence of this module is the following:

* Common library/package to provide capabilities to read, image data from a product and to create a new tile crop mask image.
* S2Agri application package

This component should be initialized at the beginning of the module.

#### Interfaces

The main interfaces in term of control flow are:

* The job order which indicates the main information about the processing input and output data, the location of log files and the location of the working directory.

The main interfaces in term of data flow of its module are:

* As inputs:
  + a NDVI\_TS\_tile\_PRD
  + a tile pixel based crop mask: L4A\_tile\_PRD
* As output, a new L4A\_tile\_PRD
* As auxiliary data: not applicable here
* As image processing parameters:
  + The minimum object size of the segmentation

#### Resources

Not applicable here

#### References

The algorithm part of this component is described in the Crop Mask ATBD [RD.4]

#### Data

No internal data is currently planned.

1. The user should be warned that the crop type processor will be blocked until this value is not filled. [↑](#footnote-ref-2)