



Contribution Checklist

1	Contents	
2	Pre-development	3
3	General	3
4	Concept Design	3
5	Development	3
6	General	3
7	Code	4
8	Component	4
9	Concept Design	4
10	Pre-submission	5
11	General	5
12	Code	5
13	Review	6
14	General	6
15	Code	7
16	Component	7
17	Concept Design	7
18	Post-acceptance	8

19 This document covers the steps that should be taken at the various stages of
20 making a contribution to Apertis with the rationale more fully explained in the
21 [policy](#)¹. It covers both those steps to be taken by the contributor as well as the
22 [maintainer\(s\)](#)² accepting the contribution on behalf of Apertis. It is presented
23 in this manner to provide transparency of the steps that are taken and the
24 considerations that are made when accepting a contribution into Apertis, be
25 that a modification of an existing component, addition of a new component or
26 concept design.

27 The steps required to be taken by a contributor will be marked with `Contributor`,
28 those to be taken by a maintainer on behalf of the Apertis project will be labeled
29 `Maintainer`.

30 Apertis is utilized by multiple parties, all of whom have a stake in Apertis
31 continuing to meet their own set of requirements. Whilst a proposed change
32 may provide the optimal solution for your use case, the Apertis maintainers will
33 need to consider the impact the change will have on the other users of Apertis too
34 and thus may request changes to the solution to ensure that Apertis continues
35 to well serve all its users.

36 Depending on the scope and content of the requested change, options may be

¹<https://jwd.pages.apertis.org/apertis-website/policies/contributions/>
²<https://jwd.pages.apertis.org/apertis-website/policies/contributions/#the-role-of-maintainers>

37 available to provide a [dedicated project area](#)³ to host party/project specific
38 packages to enable such changes to be made to a project specific version therefore
39 avoiding and impact on the core Apertis offering.

40 This checklist is broken down into the following stages, with some items broken
41 out per contribution type:

- 42 • **Pre-development:** These are topics that should be addressed prior to any
43 significant work being carried out to avoid pitfalls that may cause a con-
44 tribution to be rejected.
- 45 • **Development:** Certain factors and considerations should be made dur-
46 ing the development of proposed changes to ensure they conform to the
47 projects polices.
- 48 • **Pre-submission:** Final checks that should be made prior to a submission
49 being made.
- 50 • **Review:** Points that should be covered during the review of the contribu-
51 tion.
- 52 • **Post-submission:** These are on-going responsibilities after a change has
53 been accepted.

54 Pre-development

55 General

- 56 • **Contributor Understand licensing requirements:** Ensure that the con-
57 tribution will be able to be [licensed in a manner acceptable to the Apertis](#)
58 [project](#)⁴.
- 59 • **Contributor Determine if ongoing support is to be provided:** Aper-
60 tis is supported with resources and effort by it's core backers. It is the
61 requirements of those who support the project who ultimately control
62 its direction. Whilst simple non-intrusive changes are very welcome, the
63 ability to offer firm commitments to support the project may impact the
64 viability of a proposed substantial change that provides no benefit to the
65 existing maintainers.
- 66 • **Contributor Identify the value that the proposed changes bring to**
67 **Apertis:** During review, the Apertis maintainers will consider the [value](#)
68 [brought to Apertis](#)⁵ by any proposed changes. Ensure that such value can
69 be expected before starting development.
- 70 • **Contributor Ensure proposed changes comply with all existing rel-**
71 **evant policies:** The Apertis maintainers will be evaluating whether the
72 proposed changes comply with the full range of policies which govern the

³<https://jwd.pages.apertis.org/apertis-website/policies/contributions/#dedicated-project-areas>

⁴<https://jwd.pages.apertis.org/apertis-website/policies/license-applying/>

⁵<https://jwd.pages.apertis.org/apertis-website/policies/contributions/#extending-apertis>

73 Apertis project. Now is a good time, before significant effort is expended,
74 to check that the proposed changes align with the Apertis projects policies
75 to guide development and avoid disappointment or wasted effort.

76 Concept Design

- 77 • Contributor **Survey the state of art:** The project strives to adapt and
78 expand to new use cases utilizing the approach that provides the best
79 fit for Apertis. Any proposed change to the project should show that
80 alternative have been researched and evaluated.

81 Development

82 General

- 83 • Contributor **Explain what the contribution brings to Apertis:**
 - 84 – Code: What does the change do?
 - 85 – Component: What is the component, what does it do?
 - 86 – Concept Design: What is the goal, how is it expected to work?
- 87 • Contributor **Any impacted documentation is updated:** This may be
88 able to form part of the same merge request or may need to be part of a
89 separate merge request depending on the repository to which changes are
90 being made. Either way, such changes should be available for review at
91 the same time.

92 Code

- 93 • Contributor **Coding conventions:** Ensure that any code conforms with
94 the [Apertis coding conventions](#)⁶
- 95 • Contributor **Changes don't break any supported architecture:** Aper-
96 tis provides support for a number of [reference platforms](#)⁷, the changes
97 must work or not be applicable on all applicable architectures and plat-
98 forms.

99 Component

- 100 • Contributor **Components should follow the packaging workflow:**
101 Repositories for newly added components should be structured accord-
102 ing to the [component structure](#)⁸ guide, including providing Apertis' CI
103 pipelines. The pipeline should succeed for all supported architectures.

⁶https://jwd.pages.apertis.org/apertis-website/policies/coding_conventions/

⁷https://jwd.pages.apertis.org/apertis-website/reference_hardware/

⁸https://jwd.pages.apertis.org/apertis-website/guides/component_structure/

104 Where a components applicability is limited to specific platforms or archi-
105 tectures, this should be well documented and the components repository
106 configured to reflect this.

107 Concept Design

- 108 • Contributor ****Follow document template****: The document should utilize
109 the [document template](#)⁹ as a framework when writing the concept design.
- 110 • Contributor **Document expected approach to meeting goals**: An
111 outline should be giving a high level overview of the steps that would
112 need to be taken to take Apertis from its current state to the end goal of
113 the concept design.

114 The breadth of topics that may need to be covered here will be highly
115 dependent on the goal of the concept document. The document should be
116 detailed enough to clearly describe the design and surrounding problems
117 to developers and project managers, but it is not necessary to describe
118 implementation details.

119 Topics that may need to be addressed include changes or impact on:

- 120 – The development workflow
- 121 – CI/CD and testing approach
- 122 – Infrastructure configuration
- 123 – Existing components
- 124 – Support of releases over their lifetimes
- 125 – Long-term maintainability of the project
- 126 – Impact on security and effect on security boundaries
- 127 – Backwards compatibility with existing feature set

128 Such topics may require a high degree of familiarity with the project to an-
129 swer. The Apertis maintainers are open to discussing goals and approaches
130 prior to a concept design being submitted. Discussing and collaborating
131 with the maintainers at an early stage is likely to prove beneficial to the
132 contributor, increasing the likelihood that the submitted design concept
133 will ultimately be accepted.

- 134 • Contributor **Website integration**: Design concepts and other equivalent
135 documentation changes are submitted as a change to the documentation
136 on the Apertis website and should also be generated by the website CI/CD
137 as a PDF to aid with review. Documents should be formatted in Mark-
138 down and follow the [relevant guidance](#)¹⁰.

⁹<https://jwd.pages.apertis.org/apertis-website/policies/contributions/#concept-design-document-template>

¹⁰<https://gitlab.apertis.org/docs/apertis-website/-/blob/master/README.md>

139 Pre-submission

140 General

- 141 • Contributor **The proposed changes should be broken down into 1**
142 **or more atomic commits**¹¹: The addition of a new concept design may
143 be presented as a single commit, however is likely that many code changes
144 should be broken down into multiple well described logical commits.
- 145 • Contributor **Document impact of changes on Apertis**: What is the
146 expected outcome in Apertis? What is it adding? What needs to change?
147 Is anything being removed? Is it expected to cause any regressions?

148 Code

- 149 • Contributor **Evaluate whether Apertis is the correct place for the**
150 **contribution**: Is Apertis the most suitable place to submit the proposed
151 changes in line with Apertis' [upstreaming policy](#)¹²?
152 In some circumstances important fixes may be acceptable for inclusion in
153 Apertis in parallel with efforts being made to upstream elsewhere, so as
154 to reduce the time taken for the changes to reach Apertis' users.

155 Review

156 General

- 157 • Contributor **Address review comments promptly and fully**: It is
158 likely that most submissions will result in feedback, be that requests or
159 questions. It is expected that a resolution should be reached for any feedback
160 prior to a submission being accepted.
- 161 • Maintainer **License suitability**: Does the contribution meet the [license](#)
162 [expectations](#)¹³ and [guidelines](#)¹⁴?
- 163 • Maintainer **Evaluate the benefits of accepting the contribution**:
164 – Does the benefit of accepting the contribution outweigh the cost of
165 maintaining the changes long-term?
166 – Does the change fit the long-term goals of the Apertis project?
167 – Does the change well with the goals and objectives of the Apertis
168 project?

¹¹https://jwd.pages.apertis.org/apertis-website/guides/version_control/#guidelines-for-making-commits

¹²<https://jwd.pages.apertis.org/apertis-website/policies/contributions/#extending-existing-components>

¹³<https://jwd.pages.apertis.org/apertis-website/policies/license-expectations/>

¹⁴<https://jwd.pages.apertis.org/apertis-website/policies/license-applying/>

- 169 • Maintainer **The goal of the change adequately explained.** For small
170 code changes a well written commit message will suffice. Larger changes
171 should be accompanied by a merge review description covering the entire
172 merge request. For concept documents, the goal should be adequately
173 explained in the document its self.
- 174 • Maintainer **Evaluate the impact on Apertis:**
 - 175 – What is the expected outcome in Apertis?
 - 176 – What is it adding?
 - 177 – What needs to change?
 - 178 – Is anything being removed?
 - 179 – Is it expected to cause any regressions?
- 180 • Maintainer **Are changes broken down into atomic commits:** Good
181 practice should be followed with regards to [commit history](#)¹⁵.
- 182 • Maintainer **Changes pass on all applicable CI/CD pipelines:** The
183 changes do not cause build regressions on any supported architecture.
- 184 • Maintainer **Evaluate impact across all supported platforms:**
185 Changes should either be applicable to all supported platforms or care
186 should be taken to evaluate that platform specific changes are made in a
187 way that other platforms are not negatively impacted.
- 188 • Maintainer **Ensure adherence to Apertis policies:** Changes to Apertis
189 should only be accepted if they adhere to all the relevant Apertis policies.

190 Code

- 191 • Maintainer **Check coding conventions:** The contribution should con-
192 form to the [coding conventions](#)¹⁶.
- 193 • Maintainer **Evaluate whether Apertis is the correct place for con-**
194 **tribution:** Is Apertis the most suitable place to submit the proposed
195 changes in line with Apertis' [upstreaming policy](#)¹⁷?

196 Component

- 197 • Maintainer **Ensure that the component doesn't duplicate exist-**
198 **ing core functionality:** Where possible adding multiple components
199 that implement the same functionality should be avoided as this increases
200 maintenance for little appreciable gain. An exception to this policy exists
201 for developer tools, especially editors.

¹⁵https://jwd.pages.apertis.org/apertis-website/guides/version_control/#guidelines-for-making-commits

¹⁶https://jwd.pages.apertis.org/apertis-website/policies/coding_conventions/

¹⁷<https://jwd.pages.apertis.org/apertis-website/policies/contributions/#extending-existing-components>

- 202 • **Maintainer Component implements the required workflow:** The
203 package repository is structured as described in the [component structure](#)
204 [guide](#)¹⁸, is correctly configured and all applicable CI pipelines succeed.

205 Concept Design

- 206 • **Maintainer Concept broadly follows concept design template:** New
207 concept designs should follow the design template where possible. Extra
208 sections may be included and sections removed where appropriate and
209 where as strong argument exists to do so.
- 210 • **Maintainer Ensure an evaluation of the state-of-the-art been per-**
211 **formed:**
 - 212 – Has a comprehensive review of alternative solutions been performed?
 - 213 – Does the proposed solution seem the best fit for Apertis? (This
214 should take the rationale for inclusion into account.)
- 215 • **Maintainer Is the approach to meeting the goals is sufficiently**
216 **clear:** Have the impact of the proposed changes been fully considered.
217 Is it understood how it will work.

218 Post-acceptance

- 219 • **Contributor Continue to support the changes that have been made:**
220 Where commitments have been made regarding support of changes to the
221 project, these should be honored.
- 222 • **Maintainer Maintain changes as long as possible:** Changes added
223 to Apertis should be maintained where possible for as long as they are
224 meaningful to the project.

225 If submitting a large patch set, consider whether it can be broken down into
226 several stages, ensuring that any feedback from reviews of earlier stages are
227 applied to subsequent ones.

228 As a rule of thumb start with a lean design/change and submit it for review as
229 early as possible.

230 You can send a new design for review to the same channels used for a [component](#)
231 [contribution](#)¹⁹.

¹⁸https://jwd.pages.apertis.org/apertis-website/guides/component_structure/

¹⁹https://jwd.pages.apertis.org/apertis-website/guides/development_process/