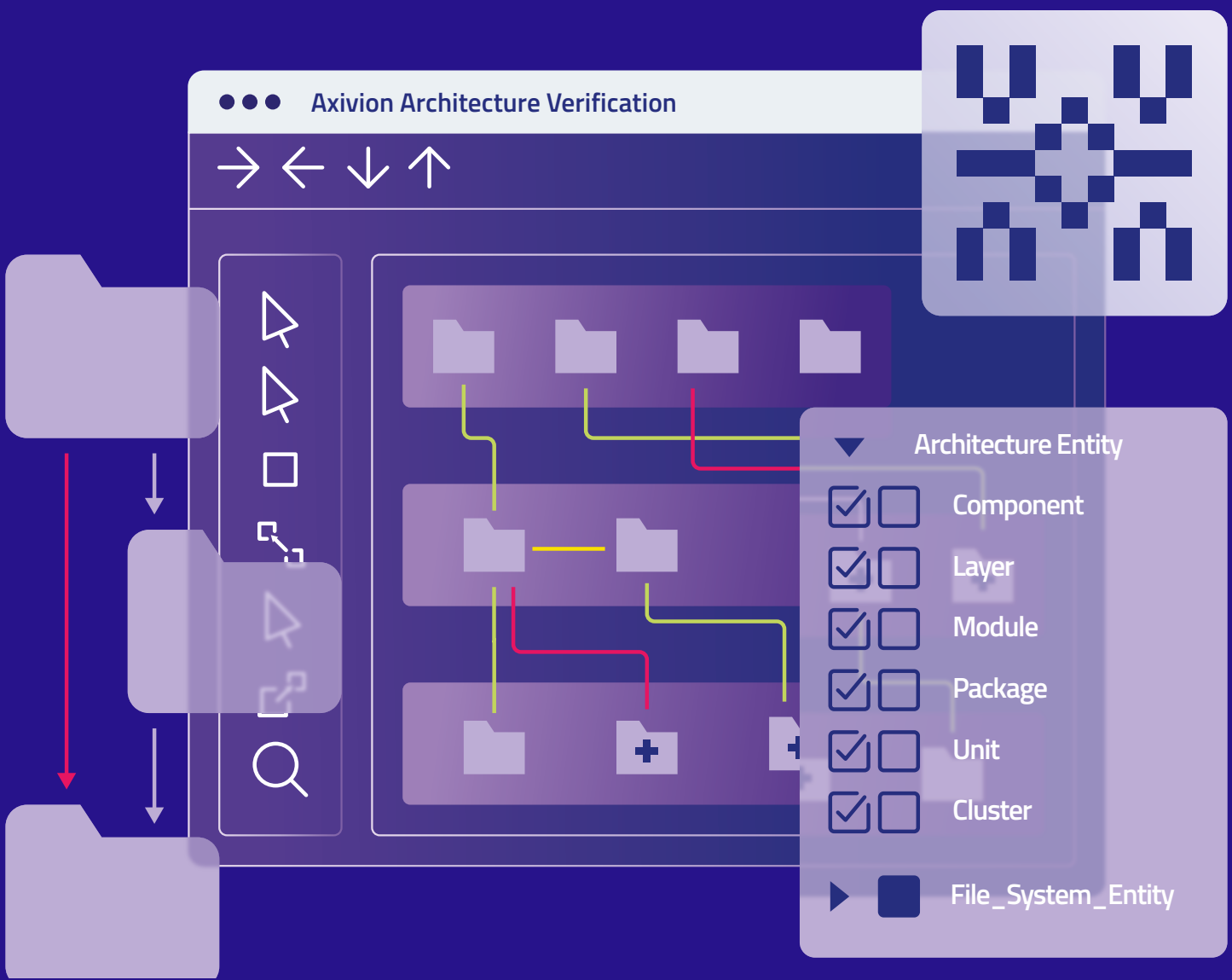
Qt Quality Assurance

Axivion Architecture Verification

如何为面向未来的软件构建可靠基础。



Axivion Architecture Verification

为何要实施架构验证?

确保代码与软件架构保持一致。

只有当软件的架构和设计与代码保持一致时，您才能确保您可以将软件架构作为讨论的指南和基线。只有这样，才能够有针对性、有计划的进行长期的产品开发。

Axivion Architecture Verification 不仅可以确保代码与架构保持一致，除功能架构之外，还可以检查并审查安全和保障架构的合规性，例如是否具备免于干扰的能力。

优势

借助可靠的文档制定关键决策。

- 使用详尽可靠的自动化架构验证取代容易出错的人工审查。
- 提取项目设计和实现的核心理念，为您的软件构建可靠的基础。
- 无需阅读代码即可了解软件架构。
- 优化新团队成员的入职培训，帮助他们从入职之初就能高效地完成工作。
- 无论是日常开发，还是规模较大的改动，比如重构代码、添加新功能等，您都能够通过架构了解任何代码变更可能产生的影响。
- 确保所有的代码更改均不违反软件架构，避免架构侵蚀（技术债务），延长软件的生命周期。

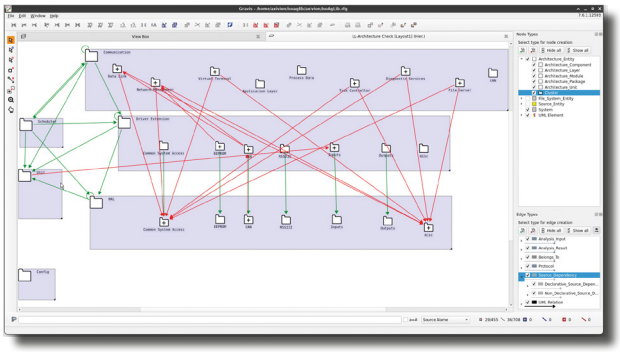
主要特性

轻松将架构验证集成到现有流程中。

- 集成建模器
- 通过 UML 工具的接口
- AUTOSAR XML 导入 (ARXML)
- 免于干扰
- 架构重建
- 针对安全和保障的架构视图

软件架构的再文档化

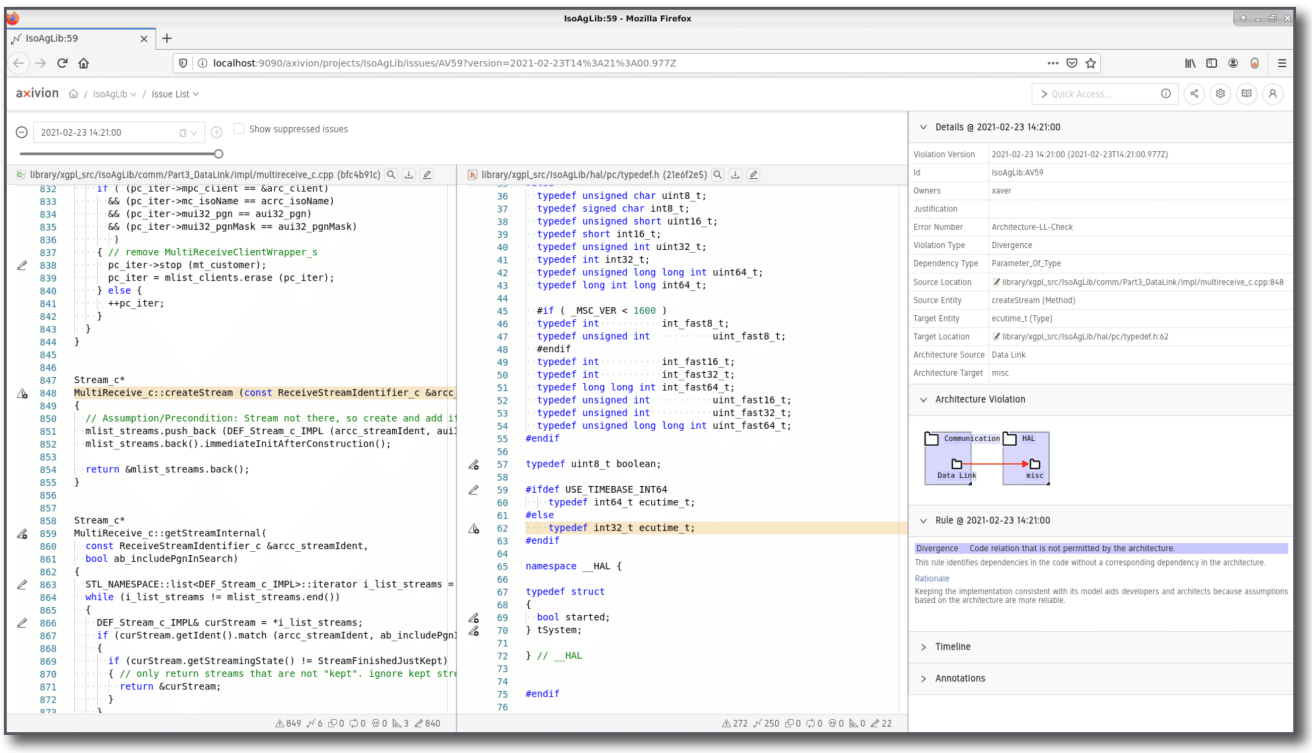
Axivion Architecture Verification 可为您的项目构建可靠的基础。无论您是否已经拥有架构模型（CASE 工具、XMI、XML 等），也无论您是只拥有一份文档（Word、PowerPoint 等），还是尚未拥有文档（不论实时项目还是后期项目），我们均能帮助您通过软件架构轻松了解软件的构建目标以及变更对软件的影响。当您需要嵌入第三方代码时，该功能将尤为有效。



与工作流程无缝集成

在引入新工具时，确保其具备易用性尤为关键。Axivion Architecture Verification 不仅可以适应您的需求，无缝集成到现有的开发环境中，还能够为您提供丰富的自定义选项，确保架构验证遵循您的规则和工作流程。在整个设置过程中，我们的资深专家将为您提供全程支持，帮助您取得超乎预期的成果。

通过每日报告，您可以轻松发现新问题，并直接定位到受影响的代码片段以进行相应的更改；借助增量分析，开发人员能够清楚区分代码问题出现的先后顺序。对时间跨度较大的内容（例如涉及多个迭代周期，甚至整个项目历史）进行审查时，智能过滤选项将帮助您依次聚焦特定类型的问题。



初始设置远比您想象得简单。

简单的设置仅需几步即可完成

借助Axivion Architecture Verification，构建和维护可靠的架构就变得轻而易举。在完成设置之后，就可以利用自动检测偏差的功能，在架构受到影响前及时修正偏差。

构建架构模型

首先要创建一个架构模型。为此，只需通过界面或集成建模器导入机器可读的现有架构模型即可。

构建代码模型

其次，从源代码中提取出代码模型，其中包括实体（如源文件）、类或函数，以及它们间的依赖关系。Axivion可通过对源代码项目的分析，自动构建相应的代码模型。

将代码映射到架构

接着，您需要将代码元素分配给架构元素，确定两者间的对应关系。根据具体的产品结构和架构模型，可选择以下映射方式：

- 手动映射（例如使用Gravis建模器）
- 使用 Axivion根据命名约定和层次信息自动映射，或
- 通过模型中给定的信息（例如标记值）自动映射

解释依赖关系

最后，您需要分别解释架构模型中的依赖关系并指明其中的含义，换句话说，即通过某种解释，使得架构依赖关系与代码依赖关系保持一致。举个简单的例子：组件A可以调用组件B中的函数。当然，尤其是对于UML模型，您还可以为依赖关系（所需/所提供的接口）提供更高级的解释。

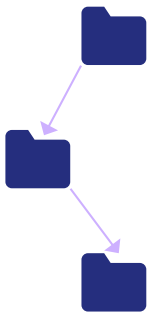
验证架构 – 每天

分析架构

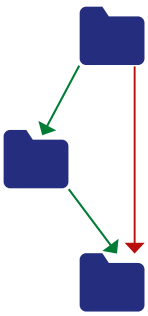
至此您已经完成了所有设置，接下来就可以使用 Axivion 进行架构验证了。在此过程中，Axivion 将分析代码并识别其中的：

从精细的架构验证中获益。

- 一致项
- 不一致项
- 缺失项



您所定义的软件架构，即理想的架构



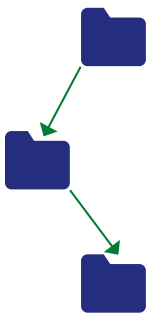
架构验证所揭示的结果，即实际的构建方式

如何处理不同的结果

在检测软件架构是否偏离预期目标时，如果想要获取详尽可靠的结果，没有什么比自动化流程更有效了。但在使用过程中，您还需要从以下选项中选择如何处理所得结果：

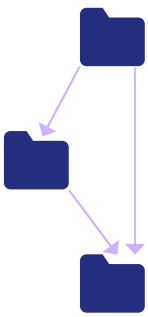
做出明智的选择。

修正源代码中的偏差



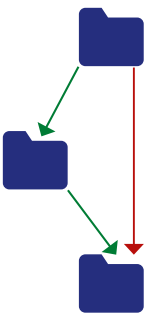
如果偏差是由错误引起的，则需修正代码以与架构重新保持一致。

更新架构规范



如果偏差确有其存在的必要性，则需更新相应的架构。

暂时接受偏差



如果偏差暂时在可接受的范围内，则可保留偏差以待后续处理。

持续分析代码

将架构验证整合到代码的持续分析中。

Axivion Architecture Verification 并非只是为当前的架构提供一次性的快照。它旨在融入整个持续的软件分析和开发流程。

完成 Axivion Architecture Verification 的设置后，您便可轻松将其集成到日常审查流程中。只需查看自动检测出的偏差并进行必要的更改，即可确保代码与软件架构始终保持一致。

使用可靠、准确的诊断信息取代容易出错的手动验证，这样不仅有助于开发人员尽早解决发现的问题，避免后期调整所需的巨额开销，同时还有助于您了解即将进行的更改会对软件带来何种影响，从而延长您构建的可靠软件架构的生命周期。

确保架构遵循 ISO 26262 标准

免于干扰

当下主流的做法是，在公用硬件上同时设置几种 ASIL 等级或 QM 等级的安全相关功能。但要实现这一点，就必须遵照 ISO 26262 标准为相应的软件项目构建适当的软件架构。因为只有遵循安全架构，才能实现免于干扰的能力。

Axivion Architecture Verification 可确保定义的接口与选择的通信机制保持一致。但凡架构中出现任何偏差，这些偏差便会立即在源文本中高亮显示，例如未指定的函数调用、数据覆盖，更常见的还有对未定义为接口的声明的引用。

基本技术规格

注：此处仅为针对 Axivion 7.8 的粗略概述。如需获取完整的规格列表，请联系我们。

支持的语言和编译器

语言	C、C++、C#

支持的操作系统

主机操作系统	Windows® 7/8/10/11、Windows® Server® 2008 R2/2012/2016/2019/2022 x86_64 GNU/Linux®（最低要求 glibc2.24 或更高版本） macOS®（最低要求 macOS 10.15）

插件

IDE	CLion、EclipseTM、基于 Eclipse 的 IDE（如 Atollic TrueSTUDIO®、CodeWarrior®、DAVETM、STM32CubeIDE、TI Code Composer StudioTM）、Microsoft® Visual Studio®、Microsoft® Visual Studio Code®、通用插件

支持的 UML® 工具

UML® 工具	IBM Rational Rhapsody、Sparx Enterprise Architect（使用 XMI 或 .qea 文件）、PlantUML

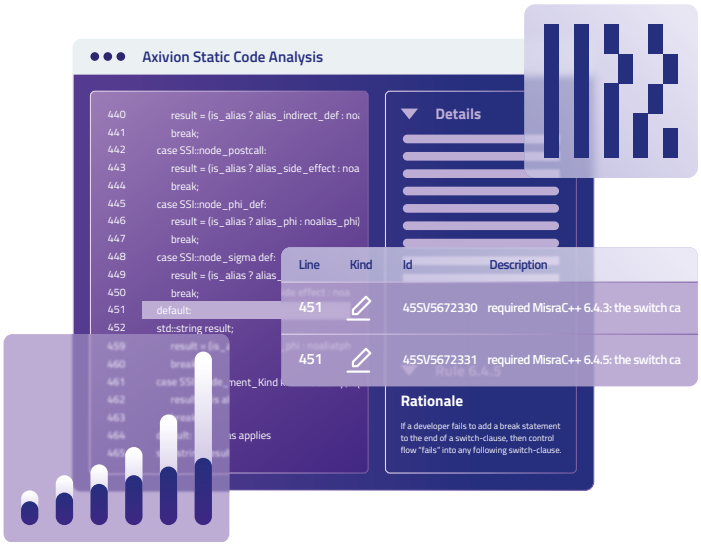
其他支持

支持的浏览器	Microsoft® Edge、Mozilla Firefox®、Google Chrome ™
要求	Python（3.8.1 - 3.12） Java® Runtime（8, 11 - 14 和 17）

就此止步？

借助 Axivion Architecture Verification，您可为软件项目构建可靠的基础；而借助 Axivion Static Code Analysis，可以对代码进行深入分析，确保代码符合最高标准。

如需了解更多内容，或希望预约会议获取演示，欢迎访问我们的网站：www.qt.io/zh-cn/product/quality-assurance/axivion-suite



Qt Quality Assurance

Qt Group (Nasdaq Helsinki: QTCOM)是一家跨国软件公司，深受各行业领导者和全球 150 多万开发者的信赖，助力打造用户衷爱的应用程序和智能设备。我们帮助客户在整个产品开发生命周期中提高生产力：从 UI 设计、软件开发到质量管理和部署。我们的客户遍布 180 多个国家和地区的 70 多个行业。Qt Group 拥有约 800 名员工，2023 年净销售额为 1.8 亿欧元。欲了解更多信息，请访问：www.qt.io/zh-cn/

www.qt.io/zh-cn/product/quality-assurance/axivion-suite