Architectural Record Guidelines: Physical Application Components (PACs)

Last Published: April 2024 Check GEAR for Latest Version

Purpose and Scope

The purpose of this document is to provide guidance on how to maintain and interpret architectural records relating to applications and the components related to those applications (plugins, addons, modules). The formal designation for an architectural record of this type is a "Physical Application Component" (PAC), but they are more commonly referred to simply as "applications" or "apps."

⚠ In the interest of readability, the term "application" will often be used as shorthand for Physical Application Component (PAC) and can be assumed to apply to both a standalone application and components (plugin, addon, module) of more complex application.

The intended audience of this document is Griffith Staff with a responsibility to maintain or interpret architectural records, which includes Enterprise Architects, Solution Architects, Technical Leads, and Product Managers.

Location of Records

Architectural Records should be maintained in Orbus Infinity:

Orbus Infinity

Orbus Infinity is the University's chosen architectural modelling tool and is part of the Griffith Enterprise Architecture Repository (GEAR):

Griffith Enterprise Architecture Repository (GEAR)

What is an Application (Physical Application Component)?

Architectural Records in Orbus Infinity and the Griffith Enterprise Architecture Repository (GEAR) typically fall into one of two main categories: Applications (Physical Application Components) and Technologies (Physical Technology Components).

To keep it simple, Griffith University defines an application as a group of functionalities that end users see as a single unit that they interact with (such as a single website or a single mobile app).

By contrast, the University defines technologies as the tools, libraries, and foundational software and hardware that is used to either deliver applications or otherwise enable staff to perform their duties.

But even with these definitions, the line between applications and technologies is not always clear, and even the most robust and complicated definitions leave room for ambiguity.

For example: While Print Drivers, Operating Systems, and Database Servers are all clear examples of technologies, what about a platform like Salesforce Marketing Cloud (SMC)? If there was a single deployment of SMC shared by the entire University, an argument could be made that it should be treated as an application, but with several different teams running their own deployments of SMC with different configurations, it would also be sensible to treat SMC as a technology and the various deployments as applications.

Ultimately, determining whether a solution should be classified as an application or technology (or both) is one of the foundational activities within any Solution Architecture, supported by the Architecture Community of Practice and reviewed by the Solution Architecture Board (SAB).

For further reference, there are several good resources online that might be of assistance in guiding the determination between application and technology:

- Martin Fowler on Application Boundaries
- Ardoq on What is an Application
- Orbus Infinity on the Core EA Model

Key Attributes

The following attributes are required for every application and are critical to reporting and governance.

Attribute	Details
Name	The name of the application as it is commonly known within the University.
Alias	A comma separated list of alternative names the application might be known by (e.g. Orbus, Orbus Infinity, iServer, iServer 365).
Description	A brief but practical description of the application and how it is used within the University.
Links	Links to more information about the application. If the application is web based, this should also include a link to the application itself.
Category	All applications should have Application selected in the category field, but there are certain edge cases where it could be useful to select additional categories. For example, there are solutions that might strictly meet the definition of Technology (Physical Technology Component) but which in practice are managed far more like applications/products. In that scenario it would be useful to select the category of Technology in addition to Application. Another scenario would be an application that is used explicitly and entirely for the purposes of accessing information. The line here is much blurrier, but if you are confident your application is information only, then it could be useful to select the category of Information in addition to Application.
Owner	The product team that helps to support, manage, or guide use of the application. Should just be the name of the team (e.g. “Reliability, Middleware & Integration” or “Student Management Solutions”). If you don’t know which product team is responsible for the application, leave it as blank or set it to “Unknown” to highlight the lack of clear ownership.
GU::Domain	The Digital Solutions Domain that helps to support, manage, or guide use of the Application.
Department	The school or department that requested the application or represent the primary users of the application. If the application is used across multiple schools or departments this should be set to Enterprise, unless there is a more accurate value that aligns to the University’s formal structure (e.g. an application that is used across multiple schools within a single faculty).
GU::Solution Classification	The Solution Classification as established by the Solution Architecture Framework and related processes.
GU::Information Security Classification	The Information Security Classification as determined by the established Solution Architecture Board (SAB) and Information Governance processes.
Vendor	The vendor who develops or manufactures the application. If the application was developed internally, this should be set to Internal.
Supplier	The commercial entity that supplied the application or through which the application is licensed. If the application was purchased directly from the vendor, this value should match the vendor value.

Attribute	Details
Application Type	Choose one of the following: • **Application**: A complete application. This should be the default if there is any doubt and will be correct for most applications. • **Component**: Many applications are now delivered in a very modular way, and it is useful to record the major components of the application separately. A simple example of this would be plug-ins for Office 365, but it would also apply to an example like the governance module of Avepoint, which is licensed separately and delivers a notable uplift in capability that is not included in the base product. A note on minor productivity applications (such as Miro or Trello) and information/consumption only websites: While minor, these are still usually applications and should be recorded as such. There may be related plug-ins that can be recorded as components, but the main products are applications. If you would like to ensure your application is flagged as belonging to one of these categories, add a “realizes” relationship to one of the following Logical Application Components: • Individual/Team Productivity Tool • Information/Consumption Only Resource
Operational Importance	An indication of how important the application is to the operations of the university: • **Enterprise Critical**: Applications that enable foundational capabilities across the entire University (e.g. Peoplesoft Campus Solutions). • **Business Critical**: Applications that enable important capabilities across the entire University (e.g. Course Profile System). • **Departmental**: Applications that enable important capabilities for a single Faculty, School, or Department (e.g. iGraduate). • **Team/Individual**: Applications that enable supporting capabilities for small teams or individuals. (e.g. H2OQ). • **External**: Applications used by the University but not owned, licensed, or managed by the University (e.g. Commonwealth Banking Platform). • **Illustrative**: Examples for architectural diagrams and models that are not actually implemented or used by the University.
Deployment Method	The best match for how the application is deployed. It is important to understand that this relates to how the application is deployed and managed, which is to say who is responsible for upgrading and supporting the application (not the platform the application runs on!). Self-Managed options imply that Griffith is responsible for upgrading and supporting the application (not the platform the application runs on!) A good example of the distinction would be our managed Wordpress solution. The Wordpress solution would be described as “Vendor Hosted (Vendor Managed)” but the applications we build using that managed Wordpress solution would be described as “Vendor Hosted (Self-Managed).” Your typical run-of-the-mill Software as a Service (SaaS) application would be described as “Vendor Hosted (Vendor Managed).”
Build	An indication of whether the application is a solution that we built (or was built for us) or solution that we bought off the shelf. Commercial-off-the-Shelf (CotS) applications are those that only require integration and configuration.

CONTINUED ON NEXT PAGE

Lifecycle & Roadmap Attributes

The following attributes relate to the lifecycle and strategic direction of the application. Lifecycle Status and Internal Recommendation are mandatory fields.

Attribute	Details
Lifecycle Status	The best fit for where the application is within a typical solution lifecycle. While some applications have regular updates, the value of In Development should be reserved for the initial development of the application. This initial development would also include configuration and integration activities for a Commercial-of-the-Shelf (CotS) application.
Internal Recommendation	The recommended target state (Tolerate, Invest, Migrate, Eliminate) of the application. Tolerate should be used as the default, with the other target states reserved for decisions that have been made as part of a formal product or architecture roadmap.
Date Of Last Release	The date the application was made available to end users, or the date the last update to the application was made available to end users.
Date Of Next Release	The date the next planned update of the application will be made available to end users (if known).
Internal: In Development From	The date which the initial development activities began (see Lifecycle Status for clarification of the term development).
Internal: Live Date	The date the application was initially made available to end users.
Internal: Phase Out From	The date on which the application is scheduled to begin being phased out. This date should be decided as part of a formal product or architecture roadmap and should only be updated as part of formal updates.
Internal: Retirement Date	The date on which the application is scheduled to no longer be available to end users. This date should be decided as part of a formal product or architecture roadmap and should only be updated as part of formal updates.
Vendor: Contained From	The date a vendor plans to begin offering limited support for an application.
Vendor: Out Of Support	The date a vendor plans to stop supporting an application.

CONTINUED ON NEXT PAGE

Standardisation & Approved Use Attributes

The following attributes should be used to indicated whether an application is considered standard, and to clarify any conditions or restrictions imposed on the application when it was approved for use.

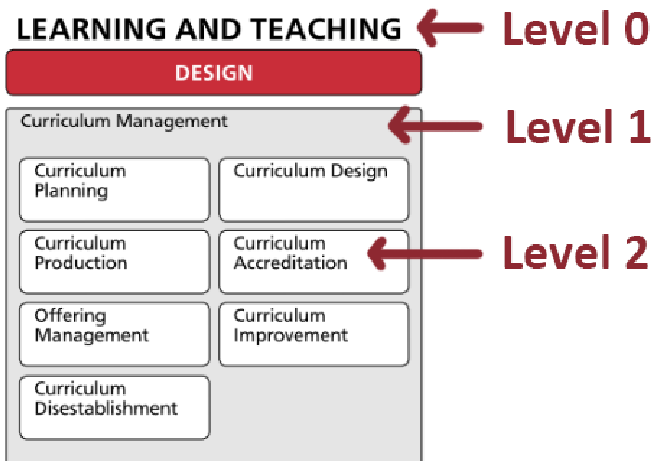
Attribute	Details
Standard Class	Indicates whether an application is considered standard, whether there are any conditions on that standard, and whether the standard is current. • **Standard**: The application is considered standard and approved for general use. • **Provisional Standard**: The application is considered standard but approved for use subject to certain conditions or restrictions. • **Proposed Standard**: The application is being evaluated for consideration as a standard and is currently approved for use subject to certain conditions or restrictions. • **Phasing Out Standard**: The application was considered standard and is approved only for continued use by users who were using the application prior to last Standard Review Date or with explicit permission from the Solution Architecture Board (SAB) or an authorised delegate. • **Retired Standard**: The application was considered a standard but is no longer approved for use without explicit permission from the Solution Architecture Board (SAB). • **Non-Standard**: The application is not considered standard but is approved for use subject to conditions and restrictions, and with explicit permission from the Solution Architecture Board (SAB).
Standard Creation Date	The date the application was first assessed for standardisation and approved use, typically the date the application was first approved to go-live by the Solution Architecture Board (SAB).
Last Standard Review Date	The date the standardisation and approved use of the application was last reviewed.
Next Standard Review Date	The date the standardisation and approved use of the application is next scheduled to be reviewed.
Standard Retire Date	The date the application was retired as a standard and approval for use was restricted.
Approved Usage	A concise summary of the uses for which the application has been approved. Further details can be provided in the Conditions & Restrictions field. For example, if an application was approved to be used by a specific researcher working on a specific research outcome, the Approved Usage field should say something like: "Approved for specific research use cases." The more specific details about which research uses cases should be provided in the Conditions & Restrictions field. This field should also attempt to highlight key boundaries for applications that have broad functionality of which only a subset has been approved. For example, if an application can be used to mock up designs and automatically generate code to implement those designs, but has only been approved to be used for mock ups, the Approved Usage field should say something like: "Approved for use in mocking up designs for mobile applications. Code automatically generated by this tool is not approved for use."
Conditions & Restrictions	A more detailed accounting of the conditions and restrictions that were imposed upon the application when it was approved for use. The description of each condition and restriction should be concise, however the intention of this field is to provide a clear and comprehensive record of any conditions or restrictions imposed on the application. For example, if an application was approved to be used by a specific researcher working on a specific research outcome, the Conditions & Restrictions field should clearly specify the research outcome, and where relevant, the person or team who has been approved to use the application.

CONTINUED ON NEXT PAGE

Key Relationships

Relationships are an extremely powerful tool to capture additional information about applications and technologies, what they deliver to the University, and how they interact.

Outlined below are a few key relationships that any application should consider, with the "Realizes Capability" relationship being mandatory.

Relationship	Target	Details
Realizes	**Capability**	The purpose of implementing applications is to enable capabilities for the University, so each application should be mapped to at least one capability. Applications should only be mapped to "Level 2" capabilities in the CAUDIT Business Capability Model.  LEARNING AND TEACHING Level 0 DESIGN Curriculum Management Level 1 Curriculum Planning Curriculum Design Curriculum Production Curriculum Accreditation Level 2 Offering Management Curriculum Improvement Curriculum Disestablishment
Communicates with	**Application (Physical Application Component)**	Applications will often interact with other applications to gather data or to trigger processes. This relationship type allows you to capture those interactions.
Is served by	**Technology (Physical Technology Component)**	Applications are often built using or serviced by an array of different technologies. This relationship type allows you to capture the technologies needed to make an application work, which can be very useful for purposes ranging from planning patches and diagnosing issues right through to long term strategic planning.