Harnessing the Power of Architectural Design Principles

We often hear people talk about the “architectural design principles” (or just “architecture principles”) of a system, but it is often unclear what they mean by this, because they haven’t defined what they mean by a “principle”. This is a lost opportunity, as defining the role of architectural design principles and the benefits of using them would allow us to capitalise on what is a very useful architectural technique.

In this column we’ll define architectural design principles, explore what a good principle looks like, and when principles might be valuable to use in architectural practice.

Defining Principles

Firstly, it’s worth stating that I’m deliberately not making a hard distinction between “design” and “architecture” because I don’t think it adds anything useful in this context. For the purposes of this discussion, we’re talking about design principles and they can apply equally well at more detailed and more abstract levels.

The Oxford English Dictionary states that a principle is “*a fundamental truth or proposition serving as the foundation for belief or action*” and that a design is a “*plan or drawing produced to show the look and function or workings of a building, garment, or other object before it is built or made*” so we can reasonably define a software design principle as:

> *A fundamental truth or proposition serving as the foundation for action with regards to deciding on the workings of a software system.*

The key point is that a principle is a clear statement of intent that guides our design work.

Other people have also tried to define design principles, notably Danny Greefhorst and Erik Proper, who have probably created the most comprehensive definition in their book “Architecture Principles” [1] when they stated that a design principle is:

> *A declarative statement that normatively prescribes a property of the design of an artefact, which is necessary to ensure that the artefact meets its essential requirements*

While a little abstract, this definition clarifies the role of the design principle, making it clear that it exists to ensure that some aspect of your architecture meets some aspect of its requirements.

So enough of abstract definitions, what does a real architecture principle look like? A simple principle based on a real example from the banking domain is shown in Table 1.

Name:	**Prefer Standardised Messaging**
Description:	When deciding on message formats between applications, we prefer: (1) To use industry standard formats (such as FpML, FIX and ISO 20022) with well documented in-house extensions where required. (2) Where this is not possible we aim to use an established organisational standard (such as EQMF). (3) Where neither is possible, a well-defined, published application-specific format should be used.

Applicability:	All application messaging for business transaction or reference data for applications with disposition “strategic”, “maintain” or “improve”.
Rationale:	• Third party formats are well thought out, standardise our workflows with the industry, make interoperation with third party software and services easier and are often handled via standard third party libraries. • Established group standards align business practice and conventions across the group, are familiar to developers, administrators and testers already and often have established libraries or processing patterns. • Where a local message format needs to be used, it should be treated like a local standard to maximise the potential for reuse and to minimise interoperability surprises. Publishing the format keeps us honest and forces us to define it properly. • All of these options may be initially more expensive than an adhoc point-to-point message format, but in the long run we know that adhoc formats are extremely expensive to maintain and limit our agility. Hence we are prepared to pay the price for favouring standardisation.

Table 1 - Example Architecture Principle

What does this principle tell us? Well it has a clear name that hints at what we’re meant do do in response to reading it, provides a short, clear description that expands on this to define it fully, and it defines its domain of applicability (messaging for certain data types for applications that have been classified in certain ways – other people need not worry about this principle). Most importantly it also explains why this is considered to be important so that we can use our judgement as to when to apply the principle. This is a useful input into the design process because it makes a priority of the organisation clear, and explains when and why it is important.

The Benefits of Architecture Principles

I think architecture principles are a useful architecture technique because they epitomise the function of architecture, which is to clearly define the necessary constraints on the design of a system, without trying to prescriptively define all the details of the design. That’s what a principle does. If you remember back to a previous column when we talked about architecture being about the “gaps” between things [2], principles help to make boundaries and priorities clear without trying to micro-manage the way that everyone does their work.

Three specific benefits that a set of architecture principles can provide are as follows:

Firstly, they can *provide a context for design decisions*. A good set of principles can help to make priorities and constraints clear and so help people make consistent and informed design decisions. In fact, I have found that they can be a good way to make quite abstract ideas like business goals clear and help designers make technical decisions that support them. We’ll come back to this idea in a bit more depth in a future column.

Next, a set of principles can be useful when needing to *justify decisions, cost and time*. As designers we are often faced with a situation where we know the right thing to do may be costly or take more time than we’d like, but it is difficult to find clear, succinct explanations of why it is the right thing to do. A set of clear principles can provide a basis for that explanation. For example, we may have a principle that all systems must be suitable for high-availability deployment, which might justify some design decisions to build capability for multi-node operation into all systems, even if this isn't the cheapest option for the immediate future.

Finally, developing a set of principles can *foster collaboration, communication and shared values*. Like many architecture artefacts, principles need to be developed by groups not by individuals, so that they are validated early and people feel collective ownership of them. Leadership is needed when defining and refining a set of principles, but defining them in a vacuum and just publishing them is unlikely to be successful. They need to be developed by the team working together and this can help people to collaborate and helps to build shared values across the group, fostering a shared understanding of what is important and what is not.

Defining Good Architecture Principles

The practical problem with principles is that they're really quite hard to define well. It is easy to come up with a long list of self-evident statements or to create long rambling statements of intent that no one really knows how to apply because they're long on philosophy and short on actionable specifics. It's difficult to produce a set which people find valuable.

My long-time collaborator Nick Rozanski has run up against this problem many times and has identified a very good list of criteria for a good architecture principle, which we included in our book [3]. These characteristics are summarised in Table 2

Constructive	A principle is stated for a definite purpose, to provide specific guidance and it is a useful guide for decision making.
Reasoned	The principle is strongly motivated by business drivers, goals, and other principles and is rational, logical and consistent.
Well Articulated	All principles need to be clearly written so that they are comprehensible by all of the necessary stakeholders.
Testable	Valuable principles are long lived and for them to be useful to needs to be possible to check objectively if it has been adhered to and if not where the exceptions are
Significant	A principle that is self-evident is rarely of value. Check that your principles are not just truisms but asking whether the the opposite statement could ever be true. If not, the principle probably isn't very valuable.

Table 2 - Characteristics of a Good Architecture Principle

These characteristics take effort to achieve but result in principles are much more likely to be valuable as they will provide significant and actionable guidance.

When to Violate a Principle

Principles are there to guide the design process and aid consistency, but it is important to recognise that architecture principles are going to get broken occasionally. Sometimes this is because people didn't realise the principle existed (perhaps because there are too many of them) or because they're just ignoring them (in which case that needs to be challenged to find out why). However, there can also be good reasons for breaking a design principle.

The principle is there to ensure consistency and alignment with goals, so breaking it has a cost, often a long term one, so when someone violates a principle, it has to be for a justifiable reason. What you need to check is that in each case, the benefits of breaking the principle outweigh the costs this implies.

However, when design principles are broken, this can provide valuable information for the architect. Firstly, the rationale for breaking the principle is valuable design information which highlights some important aspect of the system. Secondly, if a principle is (justifiably) broken routinely, then that is a strong signal that the principle needs to be changed. This can reveal a mismatch between assumptions and reality. Finally, if you know where design principles are violated in your system this is valuable information as the system evolves and you need to deal with these non-standard aspects of its design.

Conclusion

While people often talk about the architectural principles of their system they are often hard pressed to actually name them and explain their rationale. A little time spent during a system's lifecycle, particularly early in its development, to identify, debate, capture and communicate a coherent set of design principles for a system can be very valuable, for the reasons we've discussed above. It's not an easy process, but the end result helps to align theory and practice and the principles are even valuable when you find that people break them, as that in itself can provide useful information.

References

[1] D. Greefhorst and E. Propper, Architecture Principles: The Cornerstones of Enterprise Architecture, Springer, 2011.

[2] E. Woods, "Architecting in the Gaps: A Metaphor for Architecture Work," *IEEE Software,* vol. 32, no. 4, pp. 33-35, Jul-Aug 2015.

[3] N. Rozanski and E. Woods, Software Systems Architecture, 2nd ed., Addison Wesley, 2011.