What architects do

As architects we must act as a bridge between the business and technology, representing and protecting the interests of each stakeholder, often mediating between the two, but allowing the business to drive. This means keeping the following interest in balance.

- functional & quality interests
- cost containment interests
- ease-of-administration interests
- ease-of-learning
- ease-of-maintenance interests

Part of the challenge of letting the business drive is providing enough quality information about the ongoing system development effort back into the business to support good business decision making.

Goal

The ultimate goal of an architect is creating a production system while optimizing over the architectural significant requirements

Architecturally significant requirements

Architecturally significant requirements are used in system design to drive and justify decisions; if not satisfied properly, they contribute to the accumulation of technical debt. They are often non-functional requirements and are often described by quality attributes (the -ilities).

a common set of quality attributes exist, ISO/IEC 9126:

- Functionality
- Reliability
- Usability
- Efficiency
- Maintainability
- Portability

Note: ISO/IEC 9126 is een wat ouder model: nieuwe model: ISO/IEC 25010:2011

function requirements

As an architect we are the bridge between business and technology, it is our responsibility to convert the Architectural requirement to functional requirements. The functional requirements can go into a devops team.

Feedback

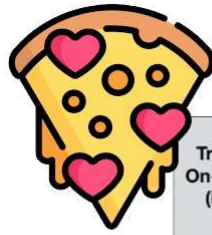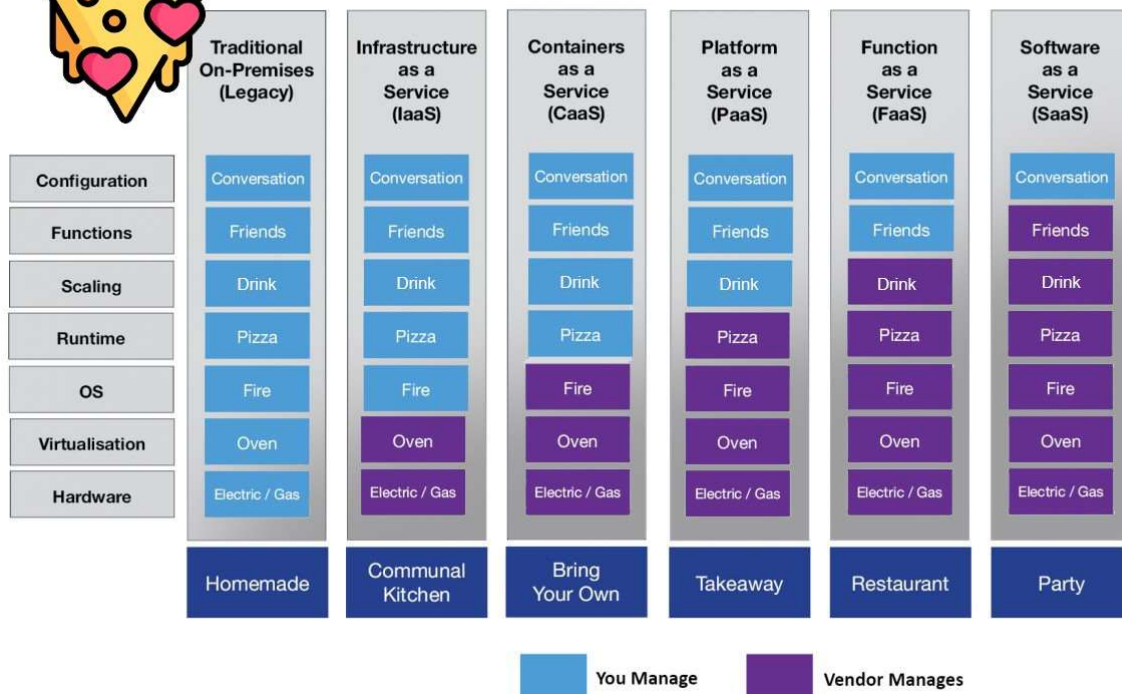Team feedback converted back to business

Team

Admins

Developers

Dependency

Dependencies on other teams can be usefull or painfull

Pizza as a Service Example

	Traditional On-Premises (Legacy)	Infrastructure as a Service (IaaS)	Containers as a Service (CaaS)	Platform as a Service (PaaS)	Function as a Service (FaaS)	Software as a Service (SaaS)
Configuration	Conversation	Conversation	Conversation	Conversation	Conversation	Conversation
Functions	Friends	Friends	Friends	Friends	Friends	Friends
Scaling	Drink	Drink	Drink	Drink	Drink	Drink
Runtime	Pizza	Pizza	Pizza	Pizza	Pizza	Pizza
OS	Fire	Fire	Fire	Fire	Fire	Fire
Virtualisation	Oven	Oven	Oven	Oven	Oven	Oven
Hardware	Electric / Gas	Electric / Gas	Electric / Gas	Electric / Gas	Electric / Gas	Electric / Gas
	Homemade	Communal Kitchen	Bring Your Own	Takeaway	Restaurant	Party

You Manage   Vendor Manages

Systems

Monitoring	nagios/libenms -> Prometheus (organizational common)
	Icinga -> checkmk (organizational common)
Provisioing	forman -> maas (makkelijker, (organizational common)
Configuration management	chef -> ansible (organizational common)
Deployment	ansible
Continues integration	bamboo (organizational enforced)
Testing	selenium
Code review	sonarcube
Build	gradle, maven, go-build, CMake
Software repository	bitbucket
Documentation	confluence -> SPHINX