2010 Seventh International Conference on the Quality of Information and Communications Technology

An IT Infrastructure Patterns Approach To Improve IT Service Management Quality

Luís Ferreira da Silva
QUASAR / CITI / FCT
Universidade Nova de Lisboa
2829-516 Caparica, Portugal
luis.silva@di.fct.unl.pt

Fernando Brito e Abreu
QUASAR / CITI / FCT
Universidade Nova de Lisboa
2829-516 Caparica, Portugal
fba@di.fct.unl.pt

Abstract — **IT services are built on top and are delivered by (and therefore depend upon) IT infrastructures. The design of the latter is critical, since it will influence the overall quality of IT services. However, designing IT infrastructures for large organizations is a challenge task since it requires knowledge of existing organization processes, the views of different players, and the conjunction of technical expertise in different domains, that rarely reside in a single individual. To improve the design of IT infrastructures, namely by allowing to reuse proven solutions to recurrent problems we propose the use of IT infrastructure patterns. The use of patterns in the design of IT infrastructure will provide several benefits such as facilitate the communication among IT design stakeholders, simplify the whole design process and potentially decrease size and complexity, which all contribute to increase the quality of IT service management processes. This paper present the preliminary effort to build supplier-independent IT infrastructure patterns and we introduce two of them, covering aspects from its rationale to instantiation that will hopefully leverage the IT infrastructure design process and create a positive impact in the quality of IT service management processes trough proven and better designed IT infrastructures.**

Keywords **– Information Technology, IT Infrastructure, Design Patterns**

I. INTRODUCTION

In a typical organization the IT Infrastructure (ITI) represents the foundation in which multiple applications are deployed to provide IT services [1]. The primary purpose of an ITI is to support and enhance IT services, so they are the foundation upon which the business processes that drive an organization's success are based [2]. In this context the infrastructure represents the use of various components of information technology and is composed with "hardware", "software" and "networking" infrastructure. All these components should work together in order to provide high quality IT services to the end-users. The quality of IT services can be defined as "the collective effect of service performances which determine the level of satisfaction of a user with the service". In practical terms IT services quality is the customer's perception of a delivered service.

There are many aspects that can influence the quality of IT services offered to customers. Since ITI is the foundation of IT services, often problems with ITIs have impact in the services provided. Most of the problems faced with IT services are frequently the consequence of poorly designed and documented ITIs created by non ITI experts such as consultants, administrators, developers and other individuals with the only purpose of responding to the requirements of a particular business application or to quickly support new technologies or services [3]. The adoption of Enterprise Architectures methodologies [4-6] in the field of ITIs is also low. One of the facts that can contribute to this, may be the existence of more than two dozen frameworks some of which are view models and others proprietary [7]. The inexistence of detailed supplier independent solutions to common problems and the inexistence of infrastructure architects to design ITIs are other reasons that highly contributes to have poorly design ITIs what often compromise the delivery of high quality services [8].

Methodologies such as IT Infrastructure Library (ITIL) [9] and Common Objectives for Information and related Technology(COBIT) [10] promote the use of a *service centric vision* for IT operations. In this *service centric vision* the IT services are measured based upon service models where services are defined based on the business processes, not based on a technology or an application. These services are monitored and maintained to meet the agreed service levels between organizations and customers. However even using this approach, poor designed ITIs lead to more problems and potentially more impact on services provided to customers.

Designing ITIs for large organizations is a challenge task mainly because it requires knowledge of existing organization processes, the views of different players, and the coordination of technical expertise in three ITI domains (hardware, infrastructure software and networking) that rarely reside in a single individual.

The design of solutions is achieved in most engineering fields by using appropriate abstractions. Although often the devil is in the details, raising the level of abstraction allows practitioners to find, share and apply standardized solutions to recurrent phenomena, by only retaining the information which is relevant for a particular purpose. Wrapping up those standardized design solutions resulted in what was coined by design patterns.

The application of the design patterns concept in the area of ITIs was caught as a business opportunity by several companies to standardize typical ITI building blocks based on

978-0-7695-4241-6/10 $26.00 © 2010 IEEE
DOI 10.1109/QUATIC.2010.34

their commercial components. Some of those companies developed methodological approaches to ITI pattern-based design, by proposing design "blueprints" embodying vendor-specific components [11, 12].

In this paper we will report our preliminary effort to build supplier-independent infrastructure patterns that can contribute to (i) improve service management quality trough proven and better designed ITIs, (ii) contribute to simplify the whole ITI design process and potentially decrease size and complexity, (iii) simplify communication among stakeholders [13], provide better integration among ITI components from different vendors and contribute to document ITIs and improving existing ITI knowledge among stakeholders. This effort is grounded on the hard-won lessons learned by the first author during several years of full-time work designing ITIs for large companies, such as banks, telecoms and big wholesale resellers.

II. Design Patterns

The work of the famous architect Christopher Alexander and his colleagues, that created the concept of design patterns, focused not only in individual patterns, but also in the concept of pattern language [14]. This term originally was meant to describe a vocabulary of interacting design strategies that can be used to develop human-scale, enjoyable and durable spaces, buildings, landscapes and towns.

Generally speaking, a pattern language is a practical network of important, related ideas that provides a, as comprehensive as possible, treatment of a subject, using a common vocabulary and understanding. Usually, such languages are the result of accumulated experience and practice [15] and can be used in various situations such as to facilitate communication, sharing of ideas, build complex and heterogeneous solutions, identify recurrent problems, and provide a guided approach to solve those problems [16], therefore improving design quality and efficiency [17]. Since we are concerned with a specific knowledge area, we propose the following definition of a pattern language for ITIs:

"An interconnected group of IT infrastructure design patterns that come together to create a secure, reliable, available, performant and manageable IT infrastructure."

The use of ITI design patterns can be seen as a process to simplify the ITI design process, while reducing its risk and cost through the use of well-known solutions for recurrent problems. The solutions addressed by design patterns are not intended to be static and final. In fact, they are templates that can be customized and extended. Design patterns help breaking ITI complexity into smaller modules, thus allowing architectural decisions to be taken at a higher abstraction level. Design of infrastructures using this approach makes them more robust, scalable, reliable, and maintainable. Our ITI design patterns have a further advantage – they are supplier independent. A pattern should provide information on how a specific problem can be addressed without focusing on a specific technology or vendor. Due to space constraints we only include here the description of two ITI design pattern, named Border Router Security Lockdown and Fault Tolerant Server, an example from the network and hardware domain. An example of an infrastructure software design pattern was presented at EuroPlop 2010 [18]. The full description of the whole collection of our ITI design patterns will be made available at the QUASAR website (http://ctp.di.fct.unl.pt/QUASAR/) as a technical report.

III. IT Infrastructure Design Patterns

As stated before the ITI is built of hardware, networking and infrastructure software and constitutes the foundation upon which organizations can deliver services to customers, partners and its employees. This section describes an example of an ITI network pattern "the *Border Router Security Lockdown"* and a hardware pattern *Fault Tolerant Server.* From the several ways to organize patterns [19-21] we decided to use a structure similar to GoF, since it is one of the most structured and well-known forms.

A. Border Router Security Lockdown

1) Context

This pattern applies to organizations using routers to connect internal LANs to the internet through a WAN link and intend to secure routers to minimize the impact of a security violation. Since all data traffic from and to the internet passes through these routers, a security violation could lead to network failures or even theft of data. Routers facing the Internet are typically known as edge or border routers

2) Problem

How to secure routers that connect internal LANs to the internet through a WAN link?

3) Forces

The following forces influence the solution:

- *Secure Connectivity:* The border router should provide routing traffic and allow both inbound traffic coming from the internet to the internal network and outbound traffic coming from the internal network to the internet.
- *Authorized access only*: The solution should take into consideration that border routers as well as other devices are not public devices and should only be accessed by authorized network engineering staff.
- *Extendability:* Solution should provide means to allow future growth through the addition of new devices while be consistent in terms of security.
- *Security:* The border zone router provides secure access to the internet. The solution should take into consideration that border router is normally the device more exposed to threats.

4) Solution

Apply tight, extended access control lists (ACLs) on the border routers to secure the network traffic allowed in the perimeter network and create and maintain security policy that identifies management activities.

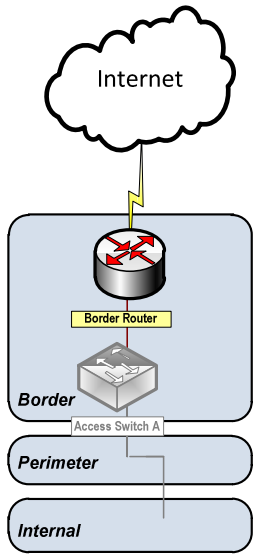

Figure 1: Border Router

The ACLs are important to restrict access from unknown, untrusted users from the internet into the internal network (Figure 1). When defining border router ACLs the following should be considered:

- Turn off unneeded services on the border router.
- Deny Internet Control Message Protocol (ICMP) to transit the border router because support for the ping command and similar capabilities can lead to potential attacks.
- Allow Border Gateway Protocol (BGP) traffic that uses TCP/179 if there are packets sourced from adjacent routers.
- Deny all protocols, TCP/UDP ports and IP addresses through the router except the protocols, ports and IP addresses corresponding to services in use.
- Block incoming packets that claim to have the same destination and source address.
- Deny tampered packets structured to appear from a different location. Anti-spoofing ACLs features ensure that tampered packets are rejected.
- Allow only internal traffic to enter the router from the internal network, and allow only internet traffic to enter the router from the external network.

The management activities should be defined in a security policy. The security policy should include:

- Change standard users and administrators names and passwords.
- Define multiple usernames and passwords for different levels of access. The policy should clearly identify who is allowed to log in to the router, who is allowed to configure and update it and who is allowed to read logs and other statistical information.
- Passwords should be complex by incorporating a mix of uppercase and lowercase letters and numbers and having the minimum length standards (for example, six characters minimum). More flexibility can be achieved by using centralized secure login system such as RADIUS or Kerberos.
- Only secure clients that use the SSH protocol and from a specific network and source IP address can establish router console sessions.
- Turn on the router's logging capability ensuring that includes time information, and use it to log errors and blocked packets.
- Having offline master copies of border router configuration files simplifies the process of identifying changes and the introduction of new devices with similar configurations.

5) Consequences

The use of BORDER ROUTER SECURITY LOCKDOWN pattern presents the following **benefits**:

- *Efficiency*: Disabling unneeded router services and restricting of packets passing through the router turn devices more efficient since they have less packets to process.
- *Access Control:* Only well-defined users and administrators can perform router management activities.
- *Simple configuration:* Having offline copies of router configuration can simplify the process of deployment of new devices or device reconfiguration due to a malfunction for example.
- *Reduce risk of attack:* By locking down border routers devices and actively control and monitor devices can minimize the risk of attack.

On the other hand the use of BORDER ROUTER SECURITY LOCKDOWN pattern carries several **liabilities**:

- *Less flexibility:* The deployment of services requiring new ports or protocols require changes in router configurations.
- *Management activities:* Since for security reasons the administration must be perform from specific locations the time and effort required to perform a configuration is higher;
- *Denial of service*: The process of locking down router does not minimize the risk of a denial of service attack.
- *Expensive*: The use of a centralized secure login may require the acquisition of new systems.

6) Related Patterns

This pattern is related with the following patterns:

- TIERED DISTRIBUTION: Tiered Distribution organizes the system infrastructure into a set of physical tiers to optimize server environments for specific operational requirements and system resource usage.
- BORDER SWITCH SECURITY LOCKDOWN: similar to border security lockdown this pattern defines the

security measures that should be applied to switches in the border zone to maximize security.

B. Fault Tolerant Server

1) Context

The definition of the required server hardware to support a new service, application or solution is a common problem for IT professionals, since it is a challenge to find the right hardware balance. For instance over-utilized servers result in performance degradation and loss of end user productivity while under-utilized servers can cause higher capital and operations expenses. The problem is even bigger when the application is critical for business.

2) Problem

Which hardware components should a server has to support business critical applications?

3) Forces

The following forces influence the solution:

- *Failures:* The failure of a single component should not affect server availability. The server should continue working by support services, applications or solutions.
- *Scalability:* The server should be designed to allow scaling up (e.g. adding more resources such as memory).
- *Performance:* The server should be performant to support business critical applications.

4) Solution

Use a server with redundant hardware components to minimize the impact of failures and provide free slots to allow the addition of new hardware components.

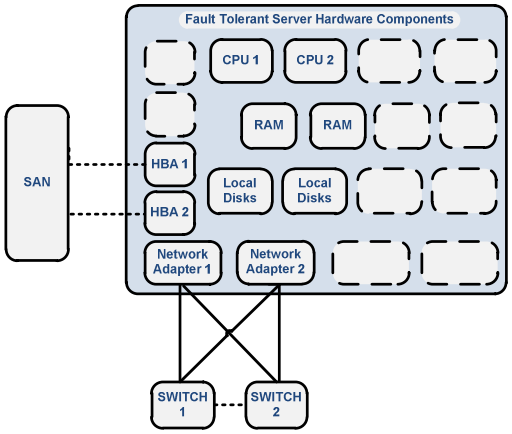

Figure 2: Fault Tolerant Server Hardware Components

The server should have the following hardware components:

- *Multiple CPUs:* In terms of CPU speed since most applications are single-threaded the number of CPUs is less important than the speed of CPUs. The faster the CPU, the more data will be able to process in a set period of time. The benefits that faster CPUs can provide to database systems for instance are obvious, however if a server has multiple slower processors, the impact of CPU speed on database systems performance can often be mitigated.
- *Memory*: The service, application or solution requirements are also the main factors to define the right amount of memory. In what concerns to memory and despite the fact that there is a correlation between number of clients to support and the total server memory, there is no ratio defined to identify the maximum amount of memory required by a server. For instance the amount of memory for a database system depends upon the previous factors and also the demands of the databases. The best practice is to configure the server with a reasonable amount of memory (depending on the requirements), and monitor whether memory is sufficient to address the server workload.
- *Network Adapters Speed*: Forgetting to define requirements for network adapters are a common mistake in the design of ITI servers since they have a significant influence on the server performance. Having the right amount of memory and faster CPUs cannot eliminate the bottleneck introduced by not having fast network adapters. Sometimes organizations perform investments to improve the performance of server through acquisition of faster processors or more memory and afterwards they realize that performance is still poor. Often the performance problems rely in NICs and not with the amount of memory or CPUs. Having a Gigabit NIC is normally recommended for most applications since these cards offer performance gains that are disproportionately higher than the 10x throughput for example.
- *Multiple Network Adapters*: The use of a multihoming (multiple network adapters) servers permits the use of teaming's which represents that a single server can have two or more network adapter ports, each connected to a physically different switch as presented in Figure 2. In case of failure of one adapter the server continues to work. Other advantages are the higher performance through the separation of traffic (e.g. traffic to internet and traffic to internal servers) and increase of security with each interface connected to a separated network segment which facilitate for instance port filtering. The disadvantages of using multihoming servers are the additional complexity, and cost associated with multiple network adapters, switches and cables.
- *Storage*: Most of applications, services or solutions also require storage to maintain information, images, files, media among others types of storage. The most common requirement is database storage. The space required can normal be segregated into the space required for database (e.g. client data), space required for processing and file storage and sometimes spaces required to backups to disk. When planning disk size, the plan should also allow for data growth over time.

Disks sized only for the initial data may quickly outgrow. Other important aspect besides space is performance often measured in throughput. The throughput can be defined as, how many requests the server can process per second. More users normally require higher throughput. In order to handle the required throughput multiple disks and disk arrays should be used or multi-channel controller cards. It is recurrent to have servers with enough memory and CPUs and with poor performance due to disk configurations. Since most applications require databases, there are two main components that a require attention when discussing disk performance for database servers, which are the database itself and database logs that should be in different disks. The common configuration to address this performance problem is to have at least three spindles, one for the operating system, one for database logs, and one for the database itself.

5) Consequences

The use of FAULT TOLERANT SERVER pattern presents the following **benefits**:

- *Availability:* The failure of most components does not affect business critical application. There are some server components (e.g. mouse, keyboard, and monitor) where a failure does not produce any direct effect in server availability.
- *Scalability:* There is enough room to scale, since there are free slots for CPUs, memory, network adapters among other components.
- *Applicability:* The unique characteristics of this server make it well suited for business application front-end servers, database servers, authentication and authorization servers among other systems.

On the other hand the use of FAULT TOLERANT SERVER pattern carries several **liabilities**:

- *Capacity Planning:* Depending on the requirements of the business application, service or solution the number and speed of hardware components may vary. The general recommendation is to use recent hardware components.
- *Single Points of Failures:* There are certain components (e.g. motherboards) that are a single point of failures in a server, since a single server cannot have more than one of these components.
- *Failure impact:* The failures of a component degrade performance.
- *Multiple failures:* The failure of multiple components simultaneously could affect server availability.

6) Related Patterns

This pattern is related with the following patterns:

- LOAD-BALANCED CLUSTER: Load-balanced clusters can improve application performance for the current number of users by sharing the workload across multiple servers.
- FAILOVER CLUSTER: Failover clusters can increase availability by creating redundancy in the infrastructure.

IV. VALIDATION

To validate the results, the plan is to use empirical research in with ITIs, which calls for the necessity of experimentation and observation rather than theory. This process consists of the application of the pattern language to real organizations. Empirical research in ITIs implies building models such as application domain or problem solving processes and checking if our understanding is correct through testing or experimenting in the real world. The analysis of results involves the ability to change or refine our models over time. As Singleton and Straits [22] stated, “appeals to authority, tradition, revelation, intuition, or other non-empirical ways of knowing, which may be acceptable in other endeavors such as philosophy, cannot be used as scientific evidence”. To better understand the impact of the solutions in ITIs, we defined several experiments and we plan to conduct these experiments with two groups. We will ask both groups to design ITIs. The first group will have access to a set of white papers and blueprints and the second group will have access to ITI patterns. We will then measure ITIs based upon the solution achieved.

V. RELATED WORK

As mentioned in section 1, some companies have proposed customized ITI design patterns, known as “blueprints”, with the obvious purpose of helping their customers to select the most adequate ITI configurations based upon their product and service offerings. Two examples that deserved our attention were the ones of Sun Microsystems [12] and Microsoft Corporation [11], which were the only ones we found with comprehensive related documentation available in the web.

Sun promotes the SDN (Service Delivery Network) approach to design service optimized network architectures for customer and in-house implementations. This approach consists of basic network building blocks, common network design patterns, integrated network components, and industry best practices that together are carefully blended in response to a customer's business and technical goals. SDN provides a set of network connectivity, routing, load balancing, and security mechanisms that, when applied in combination, result, according to [12], in “flexible network infrastructure designs that provide high performance, scalability, availability, security, flexibility, and manageability”. As for the patterns themselves, Sun proposes in the same document a set of so-called “common SDN patterns”, highlighting which key forces differentiate each pattern from the others. These patterns are based on a set of building blocks and include: the Single Service Module Pattern, the Multi-Service Module Pattern, the Single Service Module With Integration Security Module Pattern, the Single Service Module With Domain Security Module Pattern, the Single Service Module With Integration Security Module and Domain Security Module Pattern, the

Multi-Service Module With Integration Security Module Pattern, the Multi-Service Module With Service Security Module Pattern, and the Multi-Service Module With Integration Security Module and Service Security Module Pattern.

Meanwhile, Microsoft promotes a related approach, named IPD (Infrastructure Planning and Design), based on a set of guides that provide architectural guidance for Microsoft infrastructure. The current IPD documentation [23] focuses on helping the reader to plan and design the implementation of several Microsoft technologies. According to the authors, the IPD guides are supposed to assist the architect in planning for complex scenarios requiring multiple infrastructure technologies. Those guides complement product documentation by focusing on infrastructure design options and share a common structure, that includes: (i) defining the technical decision flow through the planning process, (ii) listing the decisions to be made and the commonly available options and considerations, (iii) relating decisions and options to the business in terms of cost, complexity, and other characteristics, and (iv) framing decisions in terms of additional questions to the business to ensure a comprehensive alignment with the appropriate business landscape. IPD highlights when service and infrastructure goals should be validated with the organization and provides additional questions that should be asked of service stakeholders and decision makers. Regarding design patterns, Microsoft organizes its approach in a set of design clusters including: Web Presentation Patterns, Deployment Patterns, Distributed Systems Patterns, Performance and Reliability Patterns and Services Patterns [11].

Both approaches provide methodological guidance along with ITI design patterns customized with proprietary products. Although their structure and detail are varied, both approaches can be seen as proprietary pattern languages for ITI design. We believe that a non-proprietary pattern language for ITI design like ours may play an important role in the design of ITIs comprising multiple source technologies.

VI. Conclusions

Since IT services are built and delivered on top of IT infrastructures, the quality of IT services are highly dependent on the quality of IT infrastructures. The quality of IT infrastructure starts from the design. Most critical IT Infrastructures are designed without guidelines and by non-experts what highly contributes to have complex and less reliable infrastructures what often compromises the delivery of high quality IT services. The current proprietary approaches do not comply with the use of heterogeneous (multiple source) components and most times do not represent proven knowledge. We have developed several Infrastructure Patterns as the ones presented and we are working to consolidate all the design patterns in a pattern language.

The mid-term future work is the development of tools to support infrastructure design patterns to enable patterns-driven ITI design, making the process of designing IT infrastructures more simple, more robust and providing a positive impact in the quality of IT services. We will also plan to develop ITI pattern mining algorithms, supported by a tool, both to help documenting existing legacy ITIs and to detect which are the patterns that are more used in combination.

Acknowledgment

The work presented herein was partly supported by the *VALSE project* of the CITI research center within the Department of Informatics at FCT/UNL in Portugal.

References

[1] S. Sirkemaa, "IT infrastructure management and standards," presented at the Proceedings of the International Conference on Information Technology: Coding and Computing, Las Vegas, 2002.

[2] A. Gunasekaran, *et al.*, "Performance measurement and costing system in new enterprise," 2005.

[3] R. Perry and A. Gillen, "Demonstrating Business Value: Selling to Your C-Level Executives," 2007.

[4] O. Group, *TOGAF Version 9 - A Pocket Guide*, Ninth ed. Reading - United Kindom: Van Haren Publishing, 2009.

[5] J. A. Zachman, "A Framework for Information Systems Architecture," *IBM Systems Journal*, p. 263, 1987.

[6] Gartner, "Gartner Enterprise Architecture Framework: Evolution 2005," *Gartner,* October, 25 2006.

[7] G. Booch, "Enterprise Architecture and Technical Architecture," *IEEE Softw.,* vol. 27, p. 96, 2010.

[8] R. Sessions, *Simple Architectures for Complex Enterprises*, 1 ed. Redmond: Microsoft Press, 2008.

[9] OGC. (2010, February). *Core publications for ITIL Version 3* Available: http://www.itil-officialsite.com/Publications/Core.asp

[10] ISACA. (2008, April). *Common Objectives for Information and related Technology (COBIT 4.1)*. Available: http://www.isaca.org/cobit

[11] D. Trowbridge, *et al.*, *Enterprise Solution Patterns Using Microsoft .Net: Version 2.0 : Patterns & Practices*: Microsoft Press, 2003.

[12] M. Lofstrand and J. Carolan, *Sun's Pattern-Based Design Framework: The Service Delivery Network*. Santa Clara, CA, USA: Sun Microsystems, 2005.

[13] OGC, *IT Infrastructure Library (ITIL) - Service Strategy (Version 3)*. London: The Stationery Office), 2007.

[14] C. Alexander, *et al.*, *A Pattern Language: Towns, Buildings, Construction* vol. Vol. 2. New York: Oxford University Press, 1977.

[15] F. Martin and D. Rice, *Patterns of entreprise application architecture*. New York: Addison-Wesley Professional, 2003.

[16] W. Cunningham, *The CHECKS pattern language of information integrity*. New York: ACM Press/Addison-Wesley Publishing, 1995.

[17] F. Buschmann, *et al.*, *Pattern-Oriented Software Architecture: A Pattern Language for Distributed Computing (Wiley Software Patterns Series)*: John Wiley & Sons, 2007.

[18] L. Silva, "Patterns for IT Infrastructure Design," in *Proceedings of EuroPLoP*, Irsee Monastery, Bavaria, Germany, 2010.

[19] E. Gamma, *et al.*, *Design Patterns: Elements of Reusable Object-Oriented Software*. New York: Addison-Wesley Professional, 1995.

[20] F. Buschmann, *et al.*, *A system of patterns: Pattern-oriented software architecture* vol. 1. New York: Wiley, 1996.

[21] M. Fowler. (2006, *Writing Software Patterns*. Available: http://martinfowler.com/articles/writingPatterns.html

[22] R. Singleton, *et al.*, *Approaches to social research*: lavoisier.fr, 1993.

[23] L. Dunham and P. Rytkonen, *Infrastructure Planning and Design*, 2nd ed.: Microsoft, 2009.