Internet Designs: Unveiling Simplicity

Software Systems Architecture
Group 7 – Class 3

Dinis R. dos Santos Bessa de Sousa	up202006303@fe.up.pt
João Rola Reis	up202007227@fe.up.pt
Pedro Manuel da Silva Gomes	up202006322@fe.up.pt
Ricardo André Araújo de Matos	up202007962@fe.up.pt

As we embarked on the task of evaluating our Internet architecture against the established design, we turned to the paper “Extracting The Essential Simplicity of the Internet”. Our journey begins with an exploration of the design choices that have contributed to the success of the Internet. Additionally, we reflect on how this paper has deepened our understanding of Internet principles, offering valuable insights into its functioning. We then delve into the intriguing title of the paper. Finally, we draw comparisons between our conceptual design and the intricate realities of the real-world internet architecture, illuminating both the strengths and limitations of our approach.

Internet success is set on the basis of a few essential design choices, a service model, a four layer architecture, and three mechanisms (routing, reliability, and resolution). Another two important decisions that set the path to success of the internet were the **modesty** and **modularity** of the internet.

Modesty in design entails forgoing the attempt to fulfill every conceivable application requirement in favor of embracing a more generalized service. This approach enables new applications to customize service communication according to their specific needs, facilitating faster and simpler communication.

Modularity is closely intertwined with the decision to implement a four-layer architecture. It facilitates the segregation of concerns among the layers, enabling them to evolve independently and swiftly. This modularity also permits the simultaneous operation of multiple protocols at layers 1, 2, and 4, without requiring upper layers to be aware of those beneath them; they only need access to the upward interface of their respective lower layer. For this system to function effectively, layer 3 protocol interfaces must be unique and compatible with layers 2 and 4.

Another critical factor in the success of the internet's simple design was the **encouragement for small groups to offer their working designs**. The community then selected the most suitable design to adopt.

The article emphasizes understanding the "why" rather than just the "how". This distinction is crucial because it provides insights into the underlying motivations and principles guiding these decisions. For instance, as we explore the reasoning behind interdomain routing decisions, it becomes clear **why** traditional routing protocols such as link-state and distance-vector **are unsuitable for Autonomous Systems**. ASes prioritize privacy and policy flexibility in their routing decision, qualities that neither of the conventional protocols guarantee. The link-state routing requires transparency of internal network topology, whereas distance-vector routing lacks policy flexibility.

The solution chosen to address the traditional routing protocols involves granting ASes the freedom to implement their own routing protocols while mitigating the risk of steady-state loops. This technique enables ASes to preserve policy flexibility and privacy by allowing them to selectively advertise routes to nearby ASes based on their policies. The difficulty, though, comes in stopping steady-state loops because ASes are free to choose routes on their own and present them to various neighbors. The sharing of **path information** on the internet is the solution to this issue. The whole AS-level path to the destination is included in an AS's route advertisement to a nearby AS. The receiving AS refrains from taking the advertised route if it discovers that it is already on that path in order to prevent starting a loop. The internet maintains a loop-free steady state, regardless of the diverse policy choices made by ASes, by following this restriction named “**path vector**” implemented through the Border Gateway Protocol (BGP).

We now turn to the domain of reliable transport protocols and examine the processes that guarantee effective data transfer between networks. Although ACKs are an essential component of reliable transport, **NACKs** are also useful in some situations. NACKs add value by giving the sender **quick feedback**, even though they are neither required nor sufficient for reliable transit on their own. With this information, the sender can respond quickly and start retransmissions **without having to wait for timeouts**. By combining both ACKs and NACKs, the transport protocols can achieve a balance between **reliability and responsiveness**.

“Extracting The Essential Simplicity of the Internet” is a very interesting title, but what did the authors mean by it? Most people think understanding the Internet involves understanding the details of all its protocols. However, at its core, the simplicity of the Internet can be captured by answering three basic questions: ***What tasks should the network handle?*** (the service model) ***How should we structure the Internet?*** (the architecture) and ***what are the fundamental ideas behind the key concepts (routing, reliability, and resolution)?*** By focusing on these questions, the article unravels the essential simplicity of the Internet without getting lost in its complexities.

Now it is time to compare the actual solution with our own. After having made the exercise ourselves of trying to design the Internet as if we were in the 1970’s, we feel like we have a deeper understanding not only of “what” and “how” the Internet is implemented but also “why” things were done in the way they were done.

Take routing for example, we did not account for the presence of loops and how they can affect the routing of messages. In the paper, the solution for this problem that seems complicated is as simple as a field in the header “*that starts with an initial number set by the sending host and which is then decremented every time a packet reaches a new route*r”, a field that is more commonly known as **Time-to-Live** (TTL).

Moreover, now we can more clearly see the importance and reasoning behind the **layered architecture of the Internet.** It is true that in our attempt to design the Internet, we too have described it as layered, perhaps because even when trying to make the exercise of forgetting what we knew and designing the Internet again we cannot help but think that modularity should be in place.

Only the fact that the Internet was divided into different layers permitted the introduction of, for example, new physical layer protocols, with the introduction of new technologies (such as Bluetooth or Optic Fiber), as well as the coexistence of multiple transport protocols and multiple local network designs, and that is something that might be taken for granted but was a design decision of the most importance for the Internet to adapt and evolve.

In addition, **name resolution** was something that we failed to explore in our design and could have described. We did not focus on name resolution in our design, which is now an essential part of the Internet and one of the mechanisms that makes the Internet’s growth possible. If not for the presence and use of DNS, we would have to keep a “phone book” like a document with all Internet addresses, which would not be compatible with the Internet’s constant changes.

Reading this paper has significantly enhanced our comprehension of real-world internet architecture. Armed with this knowledge, we can now discern both the strengths and limitations of our own design more clearly. For instance, we appreciate the robustness of the layered architecture, which facilitates the simultaneous operation of distinct protocols across various layers. However, we also recognize certain deficiencies, such as the absence of an application-level naming system and the potential for packets to cycle endlessly. By comparing our design with the actual internet architecture, we gain valuable insights into the complexities and nuances of designing a globally interconnected network.