"KNOWLEDGE NODES"-

FLEXIBLE INTERNET-BASED SOFTWARE SERVICE CONNECTIONS

George Simpson and Steve Boardman
Earth Observation Sciences
Farnham England
(georges@.eos.co.uk: steveb@eos.co.uk)

Abstract

It is now technically feasible to begin transition from the current "documents-only" WWW to a new generation, in which flexible distributed computational services are exchanged. We describe what that next generation might be, and discusses some of the ramifications.

1. Introduction

This paper puts forward a concept for a new stage in the development of the Internet. It sketches the main functional features, and shows that the requisite technical capabilities are already in hand. It goes on to demonstrate the implications of the concept for software development. Finally, a more speculative section suggests some interesting economic implications of the model.

We suggest that today's WWW is a mere beginning. Effectively the only use it makes of the world's computing resources is as a surrogate for books. We now can, in principle, fmd anything that is published. The problem with this is that only knowledge which has been rendered in text and graphics can be communicated. This enhances mankind's ability to learn, but not to do. In The Beginning, computers were built to execute algodthms. The current Internet's support for algorithms is minimal and this observation suggests a significant opportunity for advancement.

There are some algorithms which the Internet, through Java, does support. These mobile algorithms are small, and do not depend on significant bodies of data. We argue that there is a large class of algorithms that are immobile, either because they do depend on large data bodies, or because they themselves are large. There are other reasons that algorithms may be immobile: for example they may be under continuous evolution, or depend on special hardware, or be implemented in immobile languages such as Fortran, C, or C++. The case can be made that the majority of the most interesting algorithms are immobile.

The Object Management Group's CORBA standard offers the capability to interconnect such algorithms, is a foundation component for the concept presented here. With CORBA, algorithms written in different languages, running on different hardware, can be interconnected. This is a very important development, but we would like to see it develop one step further, as we explain below.

2. Service Concept

Consider a person with an algorithmic problem. Currently, the Internet offers such a person the ability to find documents about how the problem should be solved, and /or the ability to download software and/or data to perform the computation.

Let's take a specific example. Company X is considering where to construct a facility. They are evaluating a large number of sites in different countries around the world. They need a site near a diver, and need to know the risk of flooding at each of the different candidate sites. (This is of course just one of many considerations that will go into the site selection.)

A hydrology institute has the capability to estimate worst-case flows of water through watercourses, given a digital elevation model, surface permeability data, and rainfall statistics. Another company has the capability to construct digital elevation models, from remote sensing data. yet another company has access to most of the world's data on surface permeability, through its international connections in the water industry.

Currently, the only way for these players to co-operate in solving the original problem is through expensive and slow contractual arrangements. A tender is let to find a consultant to do the analysis, who then negotiates the required access to data and services, gathers the results into his GIS, and delivers a written report.

Wouldn't it be better if company X could find services on the web to estimate the required quantity; services that would provide digital information directly into the corporate datastream? A consultant offering such services might draw on the same kind of services from others - services to construct DEMs, deliver surface permeability for specified regions on a specified scale, and summarise rainfall records for each of the regions in the desired fashion. And below the consultant, others might offer lower-level services, such as identifying and delivering relevant satellite data.

The point is that each of these activities constitutes a complex algorithm, of the type described as "immobile" earlier. The service concept this paper presents is the capability for people to offer and use these kinds of algorithmic services.

We have coined the term "knowledge node", or K-node for short, to refer to a single algorithmic unit of this on-line service network. By using the term "knowledge", we emphasize that it is not just the algorithm, but also the people and data that support these nodes that gives them value. We will return to this point later.

Figure 1 shows how a simple vision of how a user in company X might make use of K-Nodes. A local K-node, which represents his main program, is the locus of results. This might, for example, compute a "suitability coefficient" for each site. To perform the business task he searches the WWW to find appropriate K-nodes to connect to his K-node. In the example we are considering, there might just be one, to connect to the hydrology institute that computes flood disk. Perhaps, however, the hydrology institute K-node also requires as inputs appropriate DEMs, rainfall statistics, and permeability data. The user finds suitable K-nodes that offer these services and connects them to the hydrology institute's K-node. When he clicks the "run" button, the K-nodes at each of the different sites are evoked in turn to perform the needed calculations and pass their results on, via the network.

What we have just described is the forming of a temporary "collaboration" out of pre-existing executables found on the interned

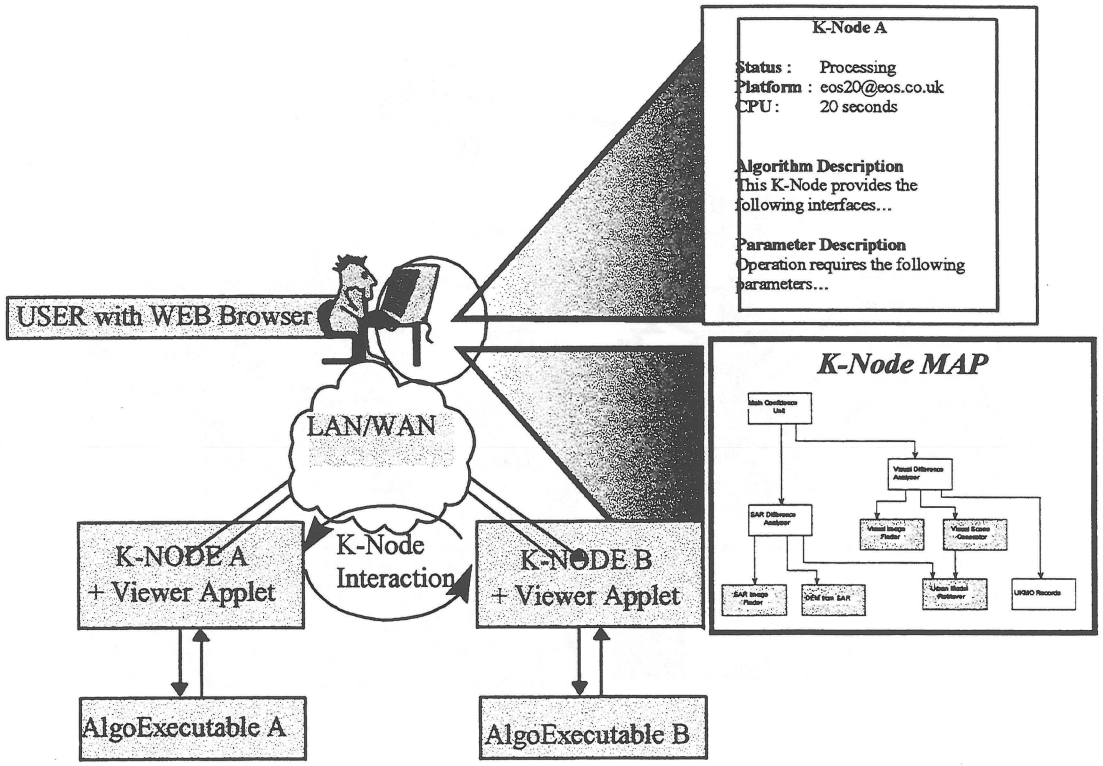

Figure 1 A user may select and configure K-Nodes into a collaboration, and execute the collaboration.

You may notice in our example that the DEM production company needs access to remote sensing data. We have not deliberately not described this part of the problem, to illustrate an important aspect of the K-nodes concept. Our user did not have to deal with this level of the problem, because the K-node the DEM company offered hid this fact - they take full responsibility for this part of the problem. It is I relevant to the user whether or not the DEM company calls on the services of other K-nodes. They may even call on services that work on a different basis.

Timing is an interesting secondary issue. It may be important for K-nodes to promise to deliver results within a specified time window.

3. Solution Technologies

WWW, Java, and CORBA are the three technologies which, when combined, make it possible to deliver this capability.

- WWW - to identify candidate service providers
- Java - to provide universally accessible connectivity
- CORBA - to interconnect the disparate forms in which the algorithms are realised.

Figure 2 shows how these technologies work together. The WWW provides search tools to identify candidate service providers, and the underlying network protocols. K-node tools written in Java offer the user an interface to select and connect K-Nodes. CORBA tools provide connections to algorithms and databases.

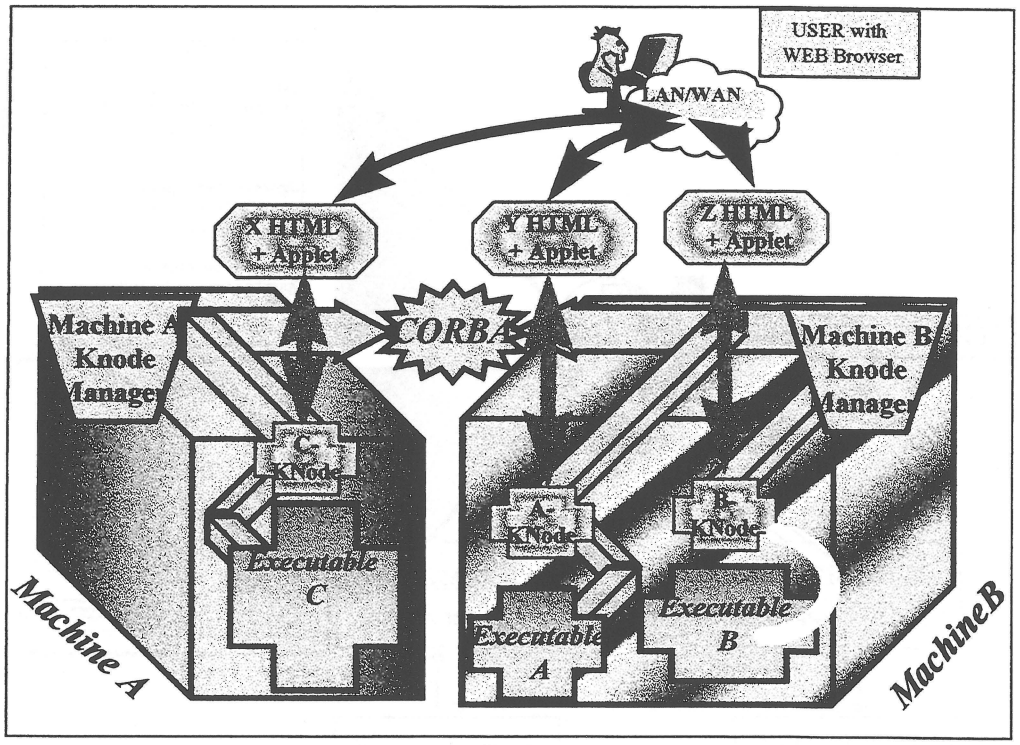

Figure 2 Three existing technologies provide most of the K-Nodes infrastructure

4. Software Development

Any set of K-nodes that work together to perform an end-user's task is a temporary collaboration, and as such represents a new and higher level of architecture. The K-nodes that are called into service for the user have no information about the overall application, and do not have to be modified in any way. The collaboration is entirely transient, and exists only for the duration of the work. (Of course, users may save the K-node network diagrams they create, and re-use them later.)

Thus the K-node concept represents a radical departure in software development, taking re-use to a much higher level. Figure 3 illustrates this. Current best practice is to build application-specific executables out of class libraries, which may be bought or bespoke. The executables are rigidly tied to the environment. This is represented in the back plane of the diagram: class libraries are strung together like beads on a string into executables. New applications are created from the same class libraries through code development.

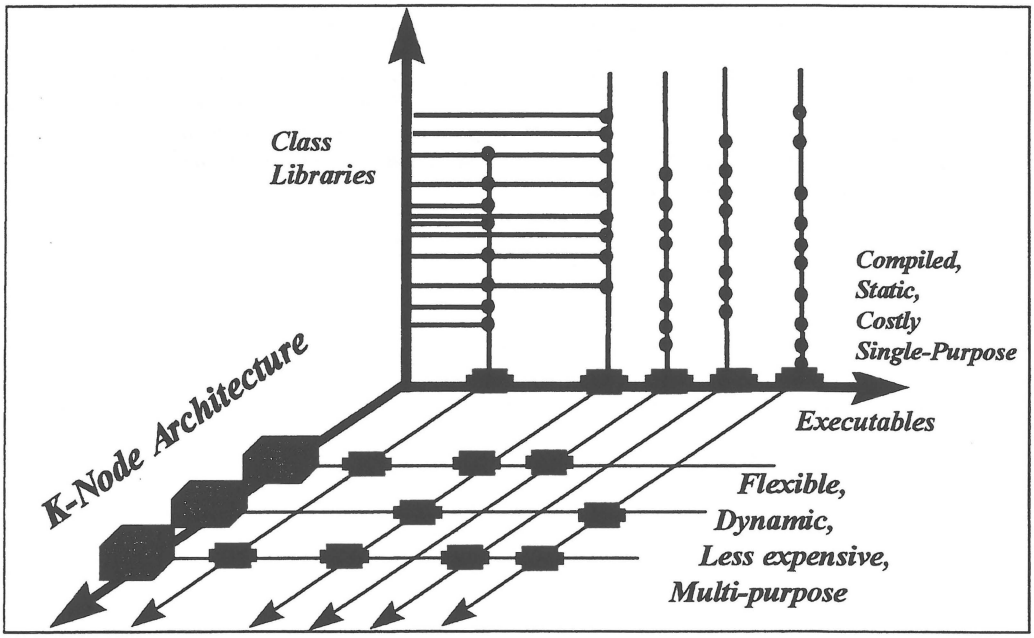

Figure 3 K-nodes takes re-use into a new dimension above classes

In contrast, K-nodes based software development makes use of existing executables in a flexible way. As mentioned above, the K-node collaborations are transient, coming into existence only for the duration of the process. Existing executables may, by offering their services via K-nodes, be used in innumerable new ways that were not conceived by their authors.

An important issue for K-nodes is specification - how to express in unambiguous and yet comprehensible terms, what something as complex as a complete executable does. Before this issue is adequately resolved, there will undoubtedly arise cases of very similar specifications arising from very disparate domains. For example, the oil and gas industry and transportation Context will have to become an important part of specifications.

However, we note that K-nodes need not, and usually would not, present the full interface of an executable. Rather various different K-nodes will be coupled to a single executable, each offering a simple interface to one of the possible services of the executable.

5. Market Aspects

How an economy of Internet-based service provision would operate is an interesting question. Certainly digital payments would play a role, but questions of liability, guarantees, and accreditation will quickly emerge. It is our view that a period of informal operation will quickly be followed by a period of standards and legislation development. Although physical distance will cease to be a factor in selecting a service provider, the strength of the legal system in the country in which a provider operates may become a factor that customers will consider.

In a networked service economy such as we are describing here, a key factor will be confidence - how sure can one be that the service being provided is what it claims to be. It is the knowledge and integrity of the individuals who create each K-node that creates and sustains this trust. Thus, unlike "intelligent agents", the K-node vision is of a computing universe in which people are very active players essentially they are the guarantors of correctness.

One interesting possibility is the emergence of a K-nodes based "cottage industry" in which individuals thrive by finding opportunities to offer high-level service packages that harness major corporate k-nodes in new ways.

6. Conclusion

EOS has prototyped the Java layer, and through a number of projects has established the viability of CORBA and WWW layers to support the concept. However, there are no plans at the moment to take the development further. This is an example of a technology that could benefit all, but which because of its inherent open-ness, offers no particular advantage to a single company.

Thus although this vision may be technically appealing, the market will determine when and if it is realised. Consider html and http, which form the backbone of the Internet today. The gap between availability of the technology and its universal availability spans more than a decade. The turning point was Mosaic, a service that made a radical improvement in the convenience of access. What will the motivating forces and turning points be in the next incarnation of the Internet, and will it look anything like K-nodes?