Decentralized Information Spaces for Composition and Unification of Web Services

POSITION PAPER

Alpa Jain and Gail Kaiser
Columbia University
Programming Systems Laboratory
Department of Computer Science
New York, NY 10027
{ajs248|kaiser}@cs.columbia.edu
August 26, 2002

The DISCUS (Decentralized Information Spaces for Composition and Unification of Services) project is building Columbia PSL's "proof-of-concept" realization of NICCI. NICCI (Network-centric Infrastructure for Command, Control and Intelligence) is a prospective DARPA program concerned with coalition forces in crisis situations; similar requirements are presented by tomorrow's enterprise applications and business services.

DISCUS focuses on rapidly forming alliances among existing legacy systems and services that may span organizational boundaries to deal with a temporary (and possibly unique) problem. The idea is that this specially-formed and dynamically-integrated suite of tools and data, termed a "Summit", will help operational teams (of humans, agents and/or applications) to achieve quick understanding and resolution of the task at hand.

A Summit is formed by composing services from different "Service Spaces", as shown in Figure 1. Each Service Space is a pre-existing collection of services allowing selective access to its member services. Being a virtual gathering of services, resources, etc. for mutual interaction and exchange, a Summit is essentially the enactment of the computation and information exchange, which could be viewed as a (virtual) distributed object. Once the capabilities collected within a Summit have accomplished their mission, the Summit can be dissolved or deactivated, although logs may be kept for postmortem analysis, and Summit "ghosts" (templates, or pseudo-classes) may be retained to even more rapidly form future Summits.

A Service Space can also be thought of as a cohesive domain in which various services, information, resources, personnel and/or legacy systems, all wrapped into Web Services[1], are placed under an administrative control. These capabilities need not be co-located, and could even span organizational boundaries, as long as there are common policies and security models. A Service Space is analogous to an Object, with Service Space templates corresponding to classes in conventional object-oriented systems.

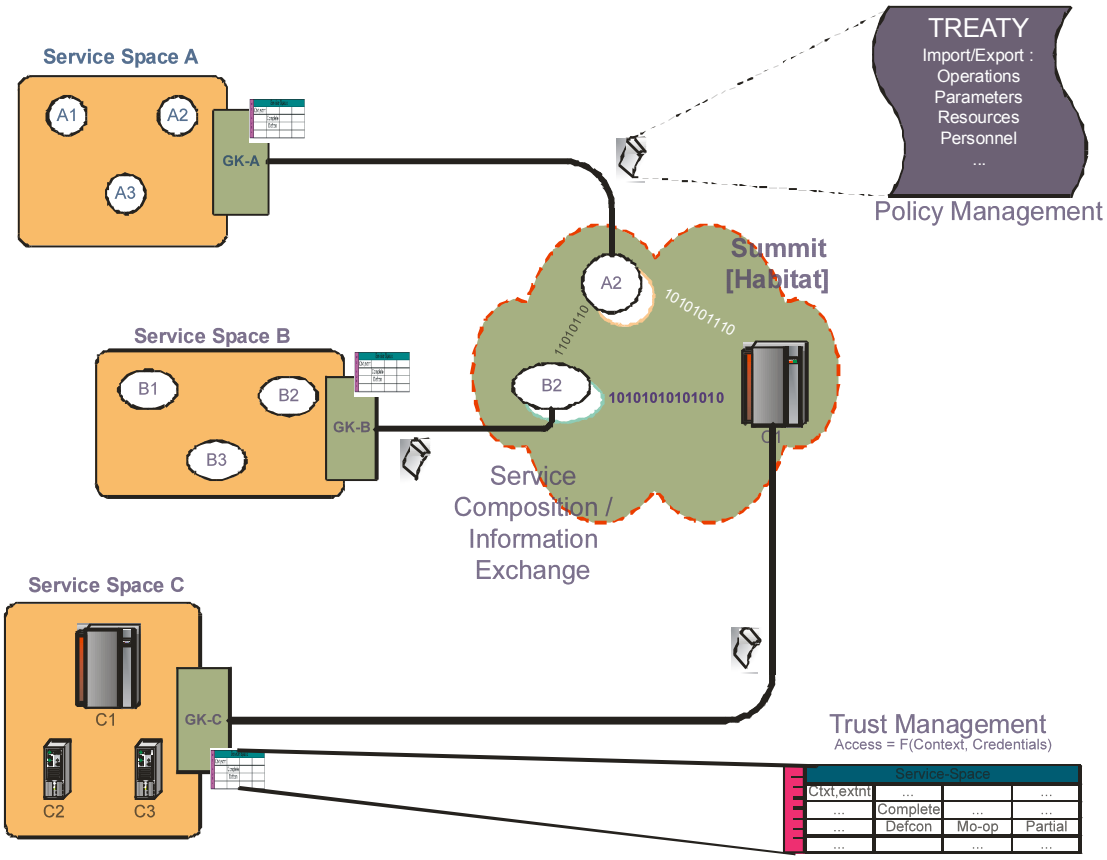

Figure 1: Overview of DISCUS architecture

Since services in a Summit interact with outside world services, external to their own Service Space's security levels and content confidentiality considerations, access to resources is likely to be restricted. The extent of access allowed by a Service Space to external entities is based on their credentials and the nature of the problem, thus requiring a dynamic trust model. Although a Service Space exports public methods and data much like a typical object-oriented class interface, it also encapsulates more complex metadata, due to the need for security and trust, service level agreements, possibly billing/payments for services rendered, etc.

A "Treaty", representing a form of contract, is dynamically formed (and reformed as needed as the mission progresses) amongst the Service Spaces of a new Summit's participating services. In essence, the services inherit the policies and constraints of their enclosing Service Space; however these are enhanced by the special circumstances reflected in the Treaty – which are now "mixed in" to the inheritance of each of the Summit's services. A Summit may then in turn act as a Service Space, making treaties with other Service Spaces and Summits in order to participate in further Summits, similar to some approaches to object aggregation.

Each Treaty is defined pair-wise between any two Service Spaces, as shown in Figure 1. The Treaty allows each of the local "Gatekeepers" to form such contracts/agreements in a fully decentralized manner without involving any global authority – just as a pair of objects might interact in a conventional object-oriented system. However, Treaty relationships are often asymmetric and never transitive: each Treaty between two Service Spaces must be formed explicitly (although a Service Space, or a Summit, may aggregate and expose selected facets of internally federated Service Spaces/Summits).

The Gatekeeper is the "traffic cop" of the DISCUS world. There is one Gatekeeper per Service Space. There cannot be any flow of data, services, etc. among Service Spaces without a Treaty. Once a Treaty has been created, any further exchange of data, services, etc. flows through the two associated Gatekeepers. Each Gatekeeper enforces normative interaction between the "enlisted" services of a Summit by intercepting and verifying every operation according to the relevant Treaty, thus enforcing the expanded object-oriented model. Authorization-related information is asserted by security matrices, allowing both matrix and hierarchical relationships among cooperating organizations and their users and internal services.

The primary enabling technology for DISCUS has been Web Services. A Web Service is a software application identified by a URI, whose interfaces and binding are capable of being defined, described and discovered by WSDL [2], making it language independent. Web Services support direct interactions with other software applications using XML-based messages via Internet protocols (SOAP) [4]. While one critical goal of DISCUS is to interoperate with multiple backend object-oriented component models (e.g., COM+, CCM, EJB), use of Web Services allows us a free choice of platforms for the infrastructure. Web Services apply web technologies such as HTTP and XML to the concepts of distributed computing technologies such as in CORBA and DCOM.

DISCUS supports:

- Quick short-term business affiliations. Summits enable automatic and transparent integration among COTS, GOTS, and open-source, etc. applications, particularly emphasizing Web Services or legacy systems wrapped as Web Services.

- Information exchange in a truly language and system independent manner. DISCUS addresses situations where multiple organizations with incompatible legacy software and possibly conflicting policies suddenly need to communicate and work together, sharing services and information.

- Dynamic membership and trust models. Every request is accompanied by the credentials of the requesting Service Space, and in some cases responses may be interrupted while in progress as membership and/or trust level changes due to changing circumstances.

- Dynamic Service Discovery. Since the service discovery mechanism in DISCUS is based on the UDDI [3] protocol, it is possible to take advantage of a variety of "advertised" software.

We have developed a "demo" application that shows rapid, secure, and simple recruitment and integration of third-party services (available through http://www.xmethods.com/) into a complex mission-oriented system by leveraging Web Services technologies. Remote services from multiple organizations are composed into a single virtual service that is then used seamlessly from within the client application (or "instigator" of the mission-at-hand). The demo scenario revolves around the task of collecting information regarding a particular location, which then becomes the basis for intelligence analyses. Additionally, information exchange with unsecured Web Services can be made secure via the Gatekeeper's interception of all data. This way, DISCUS enables the secure exchange of information among services irrespective of implementation-level details.

Future plans involve incorporating the above example into ISI's GeoWorlds [13], an integrated Geographical Information Systems and Digital Library system in use for intelligence analysis at US Pacific Command (PACOM). Traditional service integration and cooperation for such systems of systems can take weeks. DISCUS' object-oriented service composition and unification can securely assemble Web Services in seconds.

There has been research related to automated Web Service composition, addressing different issues like nomenclatures of services and resources, specification of processes and static service composition, etc. Mcllraith, et. al. [8] introduce agent-based technology into the Web Services world by proposing a DAML-enabled markup for Web Services. This markup enables automated Web Service discovery, execution, composition, and interoperation for agents and simple applications. At Hewlett-Packard, Stearns and Piccinelli [9] have applied the principles behind Separation of Concerns [7] and Aspect-Oriented Programming to develop software components for the specification and automated execution of business interaction processes based on the Web Services model. The Intelligent Web Service (IWS) System [10] presents an approach to automated interoperation amongst machines. They have developed a tool that can manipulate semantic information pertaining to services and web resources, providing a uniform representation in a machine-readable format. DISCUS is also able to incorporate automated, dynamic composition and interoperation of Web Services, yet in a target-oriented, controlled, and secure manner, assisted largely in part by the Gatekeeper entities.

In conclusion, we have presented our research work on DISCUS, which is based on the concept of temporary Summits that are rapidly and automatically assembled systems of systems that know what they are doing and why they are doing it. Summit formation and continuing operation considers the environment (including the home Service Spaces of participating systems as well as other Summits), the task's goals, and its own capabilities, taking advantage of substantial amounts of appropriately represented policies, constraints, workflows, and other knowledge. Summits form cross-organization and

multi-level security teams that succeed at missions that individual Service Spaces cannot achieve, and are robust in the face of "surprises". DISCUS builds on conventional object-oriented concepts, but departs substantially (while remaining compatible) to add distribution, security, SLAs, and other facilities necessary for large scale systems of systems – including legacy facilities or other resources not normally reflected as objects.

DISCUS is written using C# and Java, and runs on Windows .NET [14] and any other platform capable of running Java1.3 and higher. DISCUS supports pre-existing legacy facilities in any language and on any host. Additional information is available at http://www.psl.cs.columbia.edu/discus/, and the demo system can be downloaded from http://www.psl.cs.columbia.edu/software.html.

Rean Griffith and Matias Pelenur are working with Shah and Kaiser on DISCUS. The Programming Systems Laboratory is funded in part by Defense Advanced Research Project Agency under DARPA Order K503 monitored by Air Force Research Laboratory F30602-00-2-0611, by National Science Foundation CCR-9970790 and EIA-0071954, and by Microsoft Research.

References

[1] Web Services whitepaper http://www.w3.org/2002/ws/
[2] Christensen, E., Curbera, F., Meredith, G. and Weerawarana, S. (2001). Web Services Description Language (WSDL) 1.1 http://www.w3.org/TR/wsdl
[3] UDDI. Universal Description, Discovery, and Integration http://www.uddi.org/
[4] Simple Object Access Protocol (SOAP) 1.1 http://www.w3.org/TR/SOAP
[5] F. Leymann (Ed.). "Web Services Flow Language (WSFL). IBM Corp., 2001 http://www-4.ibm.com/software/solutions/webservices/pdf/WSFL.pdf
[6] S. Thatte. "XLANG: Web Services for Business Process Design" Microsoft Corp., http://www.gotdotnet.com/team/xml_wsspecs/xlangc/default.htm, 2001.
[7] P. Tarr, M. D'Hondt, L. Bergmans, and C. Videira Lopes (Ed.). "Workshop on Aspects and Dimensions of Concern: Requirements on, and Challenge Problems For, Advanced Separation Of Concerns" ECOOP'2000, Springer-Verlag, LNCS 1964, 2000
[8] S. McIlraith, T. Son, and H. Zeng "Mobilizing the Web with DAML-Enabled Web Services", The Second International Workshop on the Semantic Web (SemWeb'2001) at WWW-10. May 2001
[9] M. Stearns, G. Piccinelli "Managing Interaction Concerns in Web-Service Systems", http://aopdcs.enst-bretagne.fr/piccinelli-g-webservices1.pdf
[10] S. Suwanapong, C. Anutariya, V. Wuwongse, "An Intelligent Web Service System", http://kr.cs.ait.ac.th/Publications/webservices.pdf
[11] G. Kiczales, J. Lamping, and others. "Aspect-Oriented Programming" In Lecture Notes in Computer Science (LNCS 1241), Springer-Verlag, 1997.
[12] IBM Research: Autonomic Computing http://www.research.ibm.com/autonomic/
[13] M. Coutinho, R. Neches, A. Bugacov, V. Kumar, GeoWorlds: A Geographically Based Information System for Situation Understanding and Management
[14] Microsoft .NET http://www.microsoft.com/net/