Architectural Requirements for Computing with Coalitions of Resources

Position Paper

Mary Shaw

Computer Science Department
Carnegie Mellon University
Pittsburgh PA 15213

Mary.Shaw@cs.cmu.edu

Voice: 412-268-2589
Fax: 412-268-5576

November 3, 1998

Abstract

Widespread use of the Internet is enabling a fundamentally new approach to software development: computing through dynamically formed, task-specific, coalitions of distributed autonomous resources. The resources may be information, calculation, communication, control, or services. Unlike traditional software systems, the coalitions lack direct control over the incorporated resources, which are independently created and managed. Moreover, the resources may be transient, either because of the resource manager's actions or because of service interruptions. Development tools for resource coalitions will require new degrees of autonomy and automation in order to identify, compose, and track the resources. An economically viable reward structure will be required to establish a rich population of available resources. Evaluation will require new models of adequacy rather than classical full correctness. Computing through resource coalitions will thus create novel architectural challenges and opportunities.

Keywords: resource coalition, distributed open systems, software marketplace

Introduction

Software architecture is the study of the structures that guide the organization of software components and subsystems into complete systems. Traditionally we have studied architectures of large complex integrated systems, with emphasis on the differences among their organizational styles and especially on differences in the ways the parts interact. These examples have been closed systems—they are controlled by single institutions—as have their architectures. The constituent components may be acquired from external sources, but when incorporated in a system they come under control of the system designer. When the architectures are static they don't admit of significant runtime variation; when the architectures are dynamic they can vary only under preset constraints. Analysis of these systems uses traditional closed-system models for correctness that emphasize a priori reasoning and analysis.

Successful large systems can also emerge without central control. The Internet is such an open system: It is minimally standardized, chiefly at the level of the protocols, addresses, and representations that allow individual sites to interact. It admits of considerable variation both in the hardware that lies below these standards and the applications that lie above. There is no central authority for control or validation. Individual sites are independently administered. Individual developers can provide, modify, and remove resources at will.

With the recent exponential expansion of the Internet, especially through the access mechanisms of the World-Wide Web, a new set of architectural opportunities emerges. The Internet hosts a wide variety of resources: primary information, communication mechanisms, computation in the form of applications that can be invoked, control that coordinates the use of resources, and services such as secondary (processed) information, simulation, editorial selection, or evaluation. These resources are independently developed and independently supported. They can be composed to carry out specific tasks set by a user; in many cases the resources need not be specifically aware of the way they are being used, or even whether they are being used. The selection and composition is likely to be done afresh for each task, as resources appear, change and disappear regularly. Unfortunately, it's hard to automate the selection and composition activity because of poor information about the character of services and difficulties with interoperability and hence with establishing correctness.

Resource Coalitions: a new form of software

The shift from closed integrated systems to open resource coalitions yields a computing environment that differs from traditional environments in several important characteristics:

- The Internet provides access to a collection of resources, which are capable of invocation and interaction and are self-identifying to various degrees. It is an open system, without clear boundaries or central control.
- These resources are autonomous: they independently created and managed, and their objectives and terms of service are locally determined. They are established, modified, and removed from service by local decision. Changes may take effect without notice, even as a resource is being used.
- The resources may often be used without explicit knowledge, accommodation, commitment, or agreement of the resource manager.
- The resources are intrinsically heterogeneous: except for minimal standards for protocols, addresses, and representations, individuals and communities are free to choose their own representations, protocols, and interface specifications. No single convention or framework pervades the system. There is, of course, an incentive to use established conventions if you wish to interact effectively – but this does not preclude introducing new conventions for good (or even bad) reason.
- The resources do not automatically operate well with each other. Even if their functionality is compatible, incompatible packagings may preclude easy interoperation. Successful integration may require the integrator to provide transformation, mediation, or other glue.

In this setting, new opportunities arise for both resource providers and application developers. Some early examples already suggest the potential:

- Third-party value-added services such as resource authentication, evaluation and review. Currently in service: certificate services for software downloads such as VeriSign.
- Agents for locating resources with desired capabilities. Currently in service: search engines for Web content such as Lycos, AltaVista, HotBot and product locators such as Bottom Dollar, CompareNet, and RoboShopper.

- Agents that synthesize results from other services. Currently in service: meta-search engines such as InferenceFind and MetaCrawler.
- Distributed cooperative calculations in which a controlling machine parcels out subproblems to cooperating processors. Currently in service: large number factoring coalitions.
- Configuration evaluation services for compatibility, simulation, or performance analysis. Currently in service: tools for checking for correct versions of system extensions and of conflicts among them.

Requisite Technology

Assembling and using resource coalitions requires support beyond the usual programming environment, compiler, and link-loader. Some of the requirements arise from the need to understand the capabilities of the resources, to select and integrate them, and to validate the result. Other requirements arise from the open and dynamic character of the computing environment. Parts of the requisite technology are emerging, but it is not all in place yet.

Metainformation and credentials

Composition of components is currently difficult because it's hard to determine what assumptions each component makes about its operating context, let alone whether a set of components will interoperate well (or at all) and whether their combined functionality is what you need. Given that this is hard to do manually, you certainly can't do it automatically. I previously [Shaw96] introduced the notion of the *credential* of a component – the part of the specification that is actually documented, with each term annotated to show at least the credibility of its source.

Integration and Glue

Many useful resources exist but cannot be smoothly integrated because they make incompatible assumptions about component interaction – for example, it's hard to integrate a component packaged to interact with remote procedure calls with a component packaged to interact through shared data in a proprietary representation. Some can be integrated, but in irregular ways (e.g., VeriSign certificates, digital signatures, and digital watermarks accomplish similar ends, but the content producer and the end user must invoke them in substantially different ways). Some examples of partial self-typing exist but are not universally observed: file extensions as used by Windows systems, protocol specifications as part of URLs, conventions about the text on the first line of a text file. Ockerblooom [Ock98] used such cues in his TOM system for handling incompatibilities in data representation. The ability to rectify packaging incompatibility [Shaw95] is critical to the formation of resource coalitions.

Correctness

Traditional closed-form models for correctness must be replaced with models for "good-enough"-ness. We need a different notion of correctness—one that is incremental, progressive, and approximate. There are human limitations on the correctness of requirements, especially understanding the full generality and implications of requirements and solutions. Specifications can be no better, so it's not realistic to expect large systems to be fully verified. It follows that complex systems will require oversight, checking, and recovery. In resource coalitions, where the individual resources may change or even disappear, the problem is particularly intense. We need a new criterion to replace absolute correctness—a criterion that allows us to bring together various sources of information to establish that a system is sufficiently good for the task at hand, and to do this analysis at a reasonable cost. Among other things, this implies a need for policies governing the extent of authority granted to independent agents

Transience and mutability of resources

Use of web resources is now chiefly opportunistic, and the used resource makes no commitment to the user. Since the Internet is open and resources are locally managed, individual resources may appear, change, and disappear independently. A common source of frustration on the web, for example, is that pages disappear from under links. For resource coalitions to be useful, consumers and resources must be able to negotiate assurances about completion of tasks, stability, and availability. Resource coalitions must be able to deal gracefully with change or transience of the constituent resources. Configuration management tools, now largely tuned to closed systems, must evolve.

Agent negotiations

Some agents now exist, chiefly for acquiring information (newsbots filter news feeds, web spiders search out web pages and create indexes for use by search engines). A few shopping agents exist (e.g., RoboShopper finds web pages listing products you specify), but usually for identifying product sources rather than for fully-executing the purchase. Some auction sites allow automatic bidding (e.g., Bidmaker at OnSale.com). This is a good starting point, but the technology is not ready for unattended operation. Wiederhold's projections about mediators [Wie95] suggest a starting point.

Security

Resource coalitions require authentication of identity, privacy of content, and integrity of certain parts of the resource information. The identity may be a link to a responsible agent, not necessarily to an identifiable human. Some resources exist, such as digital signatures, digital watermarks, and certificates for identification and encryption for privacy. However, these elements are not easy to integrate with other components and may, indeed, require different forms of integration from one case to another. More detailed security requirements and appropriate responses to threats will vary from one coalition to another.

Availability is a somewhat lesser concern. Resource coalitions must be relatively resilient to service interruptions, because of the inherent autonomy of the resources.

Usability by Nonexperts

Spreadsheets, word processors, and the World-Wide Web have brought computing under the control of individuals who are not professional software developers. A key to this accessibility is that the software (a) presents a model that is easy to understand, either because it matches a model that many people already understand or because the model itself is simple and (b) has a gentle learning curve, so the effort a new user must invest is commensurate with the benefit. Everyone should be able to bring together a selection of resources that is tailored to his or her current problem. Resource coalitions don't yet have simple models, but they do show promise of a gentle learning curve.

Micro-accounting and an Open market base

Notwithstanding the enormous community of sharing and support that has been the core of Internet development, resource coalitions will not be viable for mainstream computation until all resource contributors can be rewarded. The resources may have different accounting policies: micropayments, running accounts, and subscriptions, for example. They also need the ability to charge along different lines from current norms. This requires some sort of micro-accounting. It also implies authenticated components / properties, agents, 3rd-party sources of metainformation, and reasonable measures for protecting and rewarding intellectual property

A viable micro-accounting system should provide a fertile climate for third-party value-added services. These services might provide editorial services in selecting information, evaluation services for resources, brokering of needs and resources, and so on. Storage, bandwidth, and compute cycles are free or cheap. Value now resides in information, search / retrieval, computation, editorial services, evaluations, oversight, credibility, and requirements.

Resource coalitions should thrive best in an open market. Miller and Drexler described such a market in their work on agoric systems [MiDr88], which anticipated open markets, agents for identifying resources, and issues of security and trust.

Research Opportunities

Some early examples of resource coalitions already exist. For the most part, they are established manually and must be operated according to individual, idiosyncratic, usually rigid rules. Many of the elements for task-specific resource coalitions exist in some primitive form. It's safe to anticipate that industry will develop some of the requisite technologies described above. The development will most likely be rapid but incremental.

How, then, can we identify research problems that will yield results so novel that normal industrial development will be influenced by them rather than overtaking them?

I think one key is to look for ways to unify numerous special cases, to reduce the chaos of idiosyncratic incompatible packagings, and to find good solutions to problems such as correctness and configuration management that are usually ignored – that is, to developed principled models that handle current cases but are better able to generalize to more sophisticated cases.

Here are some examples of interesting research problems:

Resource characterization through metainformation: Suppose that we can associate unforgeable tags with resources by using the World-Wide Web for distribution, <meta> tags for the metainformation, and digital signatures to ensure the integrity of both the resource and the metainformation. The set of properties can be open-ended, so new information can be added as a result of analysis, experience, or manual evaluation. How would this improve our ability to identify appropriate resources, select those of sufficient quality, and compose them into a coalition?

Sufficient correctness: What's the right way to think about whether a resource coalition is good enough? Given the mutability and transience of resources plus the impracticality of writing a complete requirement before implementation, the traditional binary criterion for correctness isn't adequate. Better would be an oversight model that defined an envelope of acceptable behavior, monitored the coalition for violations of the envelope, and took corrective action.

Even better would be a model that scaled the effort of validation the severity of failure and the degree of manual oversight available. How can we select and apply validation techniques cost-effectively?

Architectural mismatch: The Internet is riddled with resources whose functionality is tempting but whose packagings don't match. What techniques can be used to close the gap?

Configuration management: One of the most severe sources of lost time on modern computers appears to be encountering inconsistent versions of components. Special programs are available to clean up some specific types of inconsistency (e.g., conflict catchers), but they aren't entirely reliable. The mutability and transience of resources on the Internet make a solution even more essential. Could improved metainformation allow a breakthrough?

Acknowledgements

These views have emerged over several years and crystallized during my recent term as a Fellow of the Center for Innovation in Learning at Carnegie Mellon. Previous discussions with colleagues in the Carnegie Mellon School of Computer Science, especially David Garlan and other members of the Composable Software Group, have laid the groundwork. Vic Vyssotsky encouraged me to think about large software as more like city planning than like product manufacture.

Bibliography

[MiDr88] Mark S. Miller and K. Eric Drexler. "Markets and Computation: Agoric Open Systems". In B.A. Huberman (eD), *The Ecology of Computation*, Elsevier Science Publishers, 1988.

[Ock98] John Ockerbloom. Mediating Among Diverse Data Formats. PhD Thesis, Computer Science Department, Carnegie Mellon University, 1998.

[Shaw95] Mary Shaw. "Architectural Issues in Software Reuse: It's not the Functionality, it's the Packaging." Proc. Symposium on Software Reuse 1995.

[Shaw96] Mary Shaw. "Truth vs Knowledge: The Difference Between What a Component Does and What We Know It Does" *Proc. 8th International Workshop on Software Specification and Design*, March 1996.

[Wie95] Gio Wiederhold. Mediation in Information Systems. *ACM Computing Surveys*. 27(2):265-267, June 1995.