Agoric Architectual Styles for Decentralized Space Exploration

Rohit Khare, Ph.D., CommerceNet Labs
Richard N. Taylor, Ph.D., UC Irvine

CommerceNet Labs Technical Report 04-02
August 2004

Agoric Architectual Styles for Decentralized Space Exploration

Rohit Khare
CommerceNet Labs
490 South California Avenue
Palo Alto, CA
+1 (650) 289-4040
rkhare@commerce.net

Richard N. Taylor
University of California, Irvine
School of Information and Computer Science
Irvine, CA 92697-3425
+1 949 824 6429
taylor@ics.uci.edu

ABSTRACT

This position paper discusses an architectural approach to managing decentralized space exploration missions. Developing control applications in this domain is complicated by more than just the challenging computing and communication constraints of space-based mission elements; future exploration missions will depend on ad-hoc cooperation between independent space agencies' elements. Currently, the frontier of interoperability is providing communication relays, as shown in by recent Mars missions, where NASA rovers relayed data via ESA satellites.

Future mission planning envisions more extensive autonomy and integration. Examples include: taking advantage of excess storage capacity at another node, multicasting messages along several paths through deep space, or even scheduling concurrent observations of an object using several instruments at different locations. An architectural style for developing mission control applications that does not depend on positive ground control from Earth could provide (a) increased margins for space-based computing systems, (b) increased reusability by an effective build-it-for-autonomy-first strategy, and (c) avoid the single-point of failure bias in standard distributed system design approaches.

In particular, we propose combining an architectural style for decentralized applications based on the Web (ARRESTED) with *agoric computing* to apply market discipline for allocating resources dynamically among coalitions of mission elements in space. Similar approaches may have applicability in other domains, such as crisis management or battle management.

Categories and Subject Descriptors

D.2.11 [**Software Engineering**]: Software Architecture – *Domain-specific Architectures*.

General Terms

Design, Economics, Reliability

Keywords

Decentralization, Disruption/Delay-Tolerant Networking

1. INTRODUCTION

Suppose that during a fine Martian sol, one portion of a rover's onboard data storage device fails. With stereo image data flowing in, the remaining buffer space may well get overwhelmed before the next opportunity to submit diagnostic telemetry back to Earth. As a result, the final reading from a spectrometer of the sol's rock-grinding experiments might get discarded. After all, it can take up to an Earth day for ground control to react and reconfigure the mission.

One way to react to this sudden flash memory chip failure is to posit that the "price" of onboard storage has just shot up — and it's being wasted by the cameras, while the spectrometer may well be willing to pay more. We might imagine that a better autonomous reaction might be to drop the stereo pair of the image, compress it more heavily, and "evacuate" those bits to an orbiting relay satellite run by another space agency as soon as possible, in order to save space for the spectrogram.

The potential for inter-agency cooperation to optimize allocation on on-orbit and on-surface elements is already being realized [5]:

> Proximity-1, a communications protocol developed by the international Consultative Committee for Space Data Systems (CCSDS), was instrumental in the success of a recent first-ever demonstration of in-orbit communication between NASA's Mars Exploration Rover (MER) Spirit and European Space Agency (ESA) Mars Express (MEX) orbiter. [1]

2. BACKGROUND

The mechanics of architecture-driven reconfiguration at runtime continue to be actively investigated by other researchers, notably in this very workshop series [4]. Even the concept of applying internal pricing of scarce resources to drive reconfiguration has been proposed [8]. Our hope is that an economic model may be useful for designing components and connectors from the ground up, not just to select components to swap in or to gracefully degrade in the face of faults [9].

2.1 Related Work

There are two other areas of broadly related work that provide context for our proposal: *agoric systems* and *disruption-tolerant networking* (DTN).

An early[1] definition of the concept of agoric systems is due to Miller and Drexler in 1988:

> Like all systems involving goals, resources, and actions, computation can be viewed in economic terms. Computer science has moved from centralized toward increasingly decentralized models of control and action; use of market mechanisms would be a natural extension of this development. The ability of trade and price mechanisms to combine local decisions by diverse parties into globally effective behavior suggests their value for organizing computation in large systems. [7]

The earliest background on DTN appeared under the guise of InterPlanetary Networking (IPN) as popularized by Cerf beginning in 2000 [2]. The definition has been expanded to consider many risks other than latency:

> … provide network services when no end-to-end path exists through the network. The primary goal is to provide disruption tolerance by organizing information flow into bundles… DTN will result in the opportunistically leveraged connectivity and the use of multiple routes while relieving the delivery node of final acknowledgement… [3]

3. OUR PROPOSAL

Successfully exploration beyond Earth orbit will almost certainly require cooperation between many mission elements controlled by several nations — even elements 'borrowed' from other missions in progress.

While there has been substantial progress on low-level data link interoperability, it is not yet clear how resources might be allocated amongst semi-autonomous exploration elements at a coarser grain. In other words, how could one write an "application" that governed the exploration of an object in space by orchestrating many different agencies' sensor and communication resources without waiting for ground control? How could such an application be built to be resilient in the face of resource failures, very high latency, and even inadvertently malicious actions?

Financial markets inspire an emerging theory for the software architecture of such decentralized systems. While there are many examples of centralized or distributed markets in the real world (such as the NYSE and NASDAQ stock markets, respectively), the most robust ones (such as the foreign currency trading markets) must tolerate disagreement rather than relying upon a single, globally-correct value. In those markets,the power to establish an equilibrium clearing price becomes decentralized amongst all the traders themselves.

This is one reason why we are pursuing this question under the auspices of a laboratory ostensibly focused on electronic-commerce research. We believe effective strategies for autonomous exploration will require individual elements in space to consider multiple, overlapping demands on time, power, and bandwidth budgets, as well as defending against new threats from faulty or even malicious members of an exploration constellation. An effective technique may indeed be setting up a "virtual economy", and letting the 'managers on the scene' in deep space make autonomous decisions on the fly.

Our formal distinction between distributed and decentralized control differentiates our approach from other "market-like" control systems. Returning to the foreign currency analogy, rather than expecting MER and MEX to negotiate resource allocation according to a single ground-controlled interchange standard, either "NASA-bucks" or "ESA-bucks", we intend to enable each element to negotiate from an autonomous perspective and still achieve equilibrium. This approach builds-in the notion of reconfiguring resources to support reuse and adaptation of elements for future missions – but also requires robust authentication and trust management to avoid being hijacked by malfunctioning components.

4. The ARRESTED Architectural Style

Whereas traditional client/server software architectures depend on consensus — as emphasized by the so-called ACID properties of transactions *(Atomicity, Consistency, Isolation, and Durability)* — eliminating critical dependency on an Earth-bound mission control center requires new architectural styles that ensure what we term the 'BASE' properties: using only *Best-effort* networking, substituting *Approximate* estimates for exact values, managing *Self-centered* trust relationships, and ensuring *Efficient* use of the network by discarding outdated information.

A first step towards such an architectural style has an extension to the Representational State Transfer (REST) style that describes how the World Wide Web works. We have incrementally modified REST to support *Asynchronous* notification of dynamically changing data; *Routing* event notifications to interested & trusted parties; *Estimating* current

[1] Not necessarily the first: another survey [10] notes that Ivan Sutherland proposed an auction to schedule computer time in 1968.

values from cached data; and assessing multiple agencies' opinions when making *Decisions* (ARRESTED). [6]

4.1 Implications of Disruption-Tolerance

Applying ARRESTED in the context of DTN suggests developing corresponding transformations at the application-layer: how to write much more delay-insensitive, disruption-tolerant, and secure protocols (or, 'services'). One of the key abstractions in the DTN research community is the 'bundle' (as opposed to a packet). In an event-based architectural style, it may be necessary for the communication layer to be aware of logical sequences of events. While the original developer of a component may have event-driven input and output interfaces, it may not be annotated with expected sequences or common patterns that ought to be correlated and delivered together.

Another interaction is in the area of naming and addressing. In an architectural style that encourages content-based routing and transformation, it may be more efficient to transmit entire event notifications to intelligent intermediaries than to dispatch nearly-identical individually-addressed notifications to each subscriber. DTN presumes that intermediaries route based on late-bound addresses, not on dynamic queries over the entire contents of a bundle.

4.2 Implications of Agoric Control

There are at least two ways agoric control affects architects of decentralized systems. First, it may be used internally to allocate 'on-board' resources such as bandwidth, storage, battery power, and instrument tasking. It is not clear whether architectural support alone will suffice to encapsulate this: while modeling an instrument as an event-driven component that uses a lossy connector can represent the effect of 'insufficient funds,' it does not permit the component developer to directly control how much the maximum bid might be. Nor does "hiding spending" permit component developers to optimize on multiple axes. This may also prove difficult because of joint-commitments: a certain task may need to reserve a minimum amount of space *and* power in order to complete. Nonetheless, there is ample precedent that suggests agoric control for scheduling which components of an architecture are concurrently active can succeed.

The second interaction of agoric control is slightly more novel: enabling dynamic collaboration with other independent entities. While the previous examples could be viewed as a "domestic market," this additional challenge is "foreign exchange" — and without benefit of a central bank to set exchange rates.

In a thoroughly hostile environment (such as the terrestrial Internet) it may be impossible to solve this problem – decentralized micropayments is an open challenge for financial cryptography. With the relaxed assumption that space-borne elements may be considered faulty, but not malicious, it may be possible to trust other agents' claims about their 'money supply.' All the same, equilibrium is not established solely by the autonomous elements; perhaps the "reserve currency" is seeded from ground control. This external source of motivation is presumably how we can influence which missions earn priority over time.

Should these experiments succeed, we hope to proceed to explore advanced aspects of agoric control, such as futures contracts, options, and other derivative instruments. This would definitely require components to be aware of spatio-temporal constraints: a day later, the bandwidth and power may be available, but the probe may be in the wrong place to observe anything! Or, on the other hand, it could "trade" with another probe that will be in the right place later on…

That speaks to our ultimate ambition: allowing onboard mission planning software to react *in situ* to the scientific knowledge it's generating. Making such value judgments is the first step towards allowing a rover to pick out which rock to explore next on its own.

5. CONCLUSION

Our goals are to (a) enable robust interworking constellations of independently developed computing devices in space, that (b) can function intelligently and autonomously in the face of high (and possibly infinite) latency, and (c) which are resilient in the face of failures and untrustworthy behaviors.

Our approach is to leverage recent theoretical and practical advances in the understanding of decentralized systems to create application-level protocols that exploit an event-notification model , providing the necessary supporting tools and infrastructure to enable the protocols to be used.

The impact of this work will be (a) increased margins for space-based computing systems, (b) increased reusability by an effective build-it-for-autonomy-first strategy, and (c) avoid the single-point of failure bias in standard distributed system design approaches.

6. ACKNOWLEDGMENTS

We would like to thank our colleagues at the Jet Propulsion Laboratory (JPL) who developed an initial proposal for the NASA Human & Robotic Technology program with us: Peter Shames and Richard J. Doyle. We would also like to thank Jay M. Tenenbaum for references to early work in intelligent agent systems that used economic rules for planning & scheduling.

7. REFERENCES

[1] CCSDS. *Proximity-1 Communications Protocol Enables High-Speed Communication at Mars*. Press Release, 3 May 2004.

[2] Cerf, V. G. et al. *Delay Tolerant Network Architecture (draft-irtf-dtnrg-arch-02.txt)*. Internet Research Task Force (IRTF), July 2004.

[3] DARPA. BAA 04-13: *Disruption Tolerant Networking*. 2004. http://www.darpa.mil/ato/solicit/DTN/index.htm

[4] Dashofy, E., Hoek, A. v. d. and Taylor, R. N. *Towards Architecture-Based Self-Healing Systems*, in *2002 Workshop on Self-Healing Systems*, (Charleston, South Carolina, November 18-19 2002), ACM.

[5] Kazz, G. J. and Greenberg, E. *Mars Relay Operations: Application of the CCSDS Proximity-1 Space Data Link Protocol*. NASA/JPL, SpaceOps 2002 Conference, 2002. http://www.ccsds.org/documents/SO2002/SPACEOPS02_P_T5_08.PDF

[6] Khare, R. and Taylor, R. N. *Extending the Representational State Transfer (REST) Architectural Style for Decentralized Systems (in preparation)* in *ACM Trans. on Software Eng. and Methodology (TOSEM)*, 2005.

[7] Miller, M. S. and Drexler, K. E. *Markets and Computation: Agoric Open Systems* in Huberman, B. ed. *The Ecology of Computation*. Elsevier, 1988.

[8] Poladian, V., Sousa, J. P., Garlan, D. and Shaw, M. *Dynamic Configuration of Resource-Aware Services*, in *26th International Conference on Software Engineering*, (Edinburgh, Scotland, 2004).

[9] Shaw, M. *"Self-Healing": Softening Precision to Avoid Brittleness*, in *First ACM SIGSOFT Workshop on Self-Healing Systems (WOSS '02)*, (Charleston, South Carolina, November 2002), pp. 111-113.

[10] Shetty, S., Padala, P. and Frank, M. P. *A Survey of Market-Based Approaches to Distributed Computing (CISE TR03-013)*. University of Florida, August 2003.