Handing Emergent Dysfunctions in Open Peer-to-Peer Systems[1]

Mark Klein
Massachusetts Institute of Technology
m_klein@mit.edu

Yaneer Bar-Yam
New England Complex Systems Institute
yaneer@necsi.org

Abstract

Open peer-to-peer (P2P) systems - made up of dynamic collections of heterogeneous components designed and operated without centralized control - are emerging as the dominant paradigm for creating large networked software systems in a very wide range of domains ranging from military command and control to power control systems to electronic commerce. One of the major open challenges involving in making peer-to-peer systems robust and scalable is learning how to anticipate and manage their emergent dynamics. This white paper describes a plan of work aimed at addressing this challenge.

1. The Challenge: Emergent Dysfunctions in Peer-to-Peer Systems

Open peer-to-peer (P2P) systems - made up of dynamic collections of heterogeneous components without centralized control - are emerging as the dominant paradigm for creating large networked software systems in a very wide range of domains ranging from military command and control to power control systems to electronic commerce to the World Wide Web itself. The reason for this is simple: the challenges our software systems must now face are simply too large, both in scale and complexity, to be handled by hierarchical control schemes with centralized development of all key software components. In some cases political or other concerns exclude the possibility of top-down control even when it is technically feasible.

One of the major open challenges involving in making peer-to-peer systems robust and scalable is learning how to anticipate and manage their emergent dynamics. Systems made up of multiple distributed inter-dependent components can and often do produce highly dysfunctional emergent behaviors, even when the components are individually implemented in a correct and apparently reasonable way. There are many types of emergent dysfunctions, ranging from chaotic or inefficient resource allocation to non-terminating collaborative decision processes (Youssefmir and Huberman 1995) (Sterman 1994) (Hardin 1968) (Chia, Neiman et al. 1998) (Waldrop 1987) (Klein 2001). The dysfunctions a system is prone to depends on its topology and how the system elements manage shared resources.

---

[1] Working Paper ROMA-WP-2001-03. October 2001. MIT Center for Coordination Science.

To make this concrete, let us consider an example from the homeland security domain. One problem that occurred as a result of the New York attacks was massive network congestion that hampered the effort of rescue personnel to get necessary information. This congestion can be understood, at least partly, as the manifestation of several emergent dysfunctions. One such dysfunction is the “Tragedy of the Commons” (Hardin 1968). Demand-based resource sharing mechanisms, which are the current dominant approach for allocating Internet bandwidth to users, are prone to this problem, which occurs when a free or flat fee resource is degraded because no individual user has an incentive to reduce their consumption of that resource. Another likely cause of the network congestion is what we can call the “groupthink” dysfunction, wherein multiple resource consumers follow each other’s lead in over-utilizing a given resource (e.g. a particular web site) although alternate less utilized resources (e.g. other web sites) are also available. A variant of the “groupthink” dysfunction is “simultaneous jump thrashing”, wherein multiple resource consumers follow each other’s lead in jumping from one resource to another, resulting in high demand fluctuation and poor load balancing. This problem has been noted in many contexts including internet router bandwidth allocation and mobile software agents. A final example is the dysfunction called “resource poaching” (Chia, Neiman et al. 1998), wherein early but low-priority tasks take away resources (e.g. phone lines, server connections) from later but more important tasks.

Networked peer-to-peer systems are thus susceptible, as we can see, to a range of serious emergent dysfunctions that are not the result of component failures, but are rather an inherent dynamical result of local decisions made without global coordination. Some things are already known about how to handle such problems. A typical response to the Tragedy of the Commons dysfunction is to institute some kind of pricing mechanism, ideally a dynamic one that is sensitive to supply and demand fluctuations. Both the “groupthink” and “simultaneous jump thrashing” dysfunctions can be mitigated by incenting some peers to avoid resource choices made by their peers, and/or by introducing a stochastic element to their resource selections. Our understanding, however, is still in its infancy. It is still not known, for example, how to deal with the prediliction of market mechanisms to the emergent dysfunction of severe price fluctuations (Waldrop 1987). A much deeper understanding is needed of what emergent dysfunctions occur in what contexts, and how they can be avoided or resolved.

2. Our Approach

Our project will address this important challenge directly by:

- identifying what kinds of emergent dysfunctions can occur in different classes of networked peer-to-peer systems
- identifying how they can be handled (anticipated and avoided, or detected and resolved) using run-time services

These insights will be evaluated in a realistic test domain.

2.1. Identify emergent dysfunction types

Simulation-Based Analysis: We will rely heavily on simulation-based approaches for systematically identifying the different classes of emergent dysfunctions that can occur in peer-to-peer systems. While analytic techniques have proven adequate for some classes of network analysis, they typically are based on severe limiting assumptions (notably, linear relationships) that are not satisfied by complex peer-to-peer systems, so many emergent phenomena can only be explored, at present, using simulation techniques.

We immediately encounter a serious challenge, however, if we take this tack. Many of the network systems we are interested in are too large to model in complete detail. To make the problem even more challenging, it is not sufficient to model the behavior of the systems without modeling the human participants. Word of mouth between people (via email etc) can be the critical factor in creating important emergent dysfunctions such as “groupthink”. So we cannot, for our purposes, model the users of peer-to-peer systems as a mere externality captured using some simple single average demand. Open peer-to-peer systems are, in addition, generally too dynamic for complete models to be relevant for long. Another challenge concerns selecting the appropriate formalism for modeling the system. There are three main classes of simulation formalisms, which differ in terms of which primitive abstractions they use to model the system of interest:

- *parameter-based*: models system as a set of key aggregate *parameters*, inter-related by continuous mathematical (differential) models of system parameter inter-relationships. These have proven successful at modeling large-scale systems.
- *state-based*: models system as a set of *states* with discrete logic-based models of the contingent transitions between them. State-based models have proven successful at enabling detailed, often deductive, analysis of system properties such as deadlock-freeness, reachability, upper time bounds, and so on.
- *entity-based*: models system as a set of freely interacting *entities* with sometimes fairly sophisticated individual behaviors. Sophisticated techniques, derived from artificial intelligence in particular, have been developed to model the such entities.

Each of these formalisms comes from different disciplines (ranging from computer science to complex systems to physics to operations research to organizational science), and have differing, and often complimentary, strengths and weaknesses. A complex system such as a peer to peer network requires, we argue, that we use a *mix* of these formalisms for different aspects of the system, in order to adequately model its dynamics:

- *agent based models* (entity-based) to model in detail the behavior of a few key players with large impacts on the system (e.g. the news organizations). Our project team has substantial previous experience in creating sophisticated agent-based simulations (Klein and Baskin 1990)
- *systems dynamics models* (parameter-based) to model large-scale aggregate behaviors such as resource use dynamics. Systems dynamics models have been applied successfully to systems with human and software participants (Sterman 1994).
- *stochastic Petri nets* and *queuing models* (state-based) to model interactions with small to medium numbers of participants, for example to analyze the impact of the key players and aggregate resource dynamics on network congestion (Murata 1989).

Simulation models to date, however, have almost exclusively been based on a single behavioral formalism, uniformly applied. How can we integrate multiple modeling formalisms in a way that proves effective for understanding emergent dysfunctions?

Multi-Scale Representations: The discipline of "multi-scale representations" has emerged to help respond to these challenges (Binder and Plazas 2001). Traditional methods for modeling complex systems either identify *ad hoc* large-scale parameters or build fine-scale models that, it is hoped, will capture all the relevant information. In practice, however, relevant parameters are often neglected, while irrelevant parameters (at too fine a scale) are included, inordinately slowing simulations. It can take many generations of models before an adequate and efficient one has been constructed. One example of this can be seen in the context of a project to develop a large scale model of traffic flow in urban environments, based upon detailed demographics and car-by-car simulations (Laboratory 2001). Traffic patterns are sensitive to small scale features like pot holes or changes in curb structure in central city locations, but are insensitive to the presence or absence of entire neighborhoods in other parts of the city. This problem can be mitigated by combining models at different scales, highly detailed ones for critical regions, and more aggregated models for the remainder. The challenge then becomes determining which parameters should be modeled, at which level of aggregation, for which sub-parts of the simulated system. This problem is complicated by the issue of non-linear sensitivity: in non-linear systems such as peer-to-peer networks, small perturbations can lead to large effects, while large perturbations can have small effects.

Members of our project team, as well as other researchers, have been developing multi-scale techniques to address these issues by extending the notion of the Renormalization Group, drawn from physics (Binder and Plazas 2001). A Renormalization Group transformation is usually applied to determine the relevant operators in the scale invariant limit of a physical theory. It is computed by repeatedly applying aggregating transformations (such as averaging) to larger and larger scales. The larger scale variables thus encode smaller scale states indistinguishable at the larger scale. Using such an approach, a magnet can be modeled, for example, as consisting of small locally interacting magnetic domains grouped together to produce larger domains whose magnetization is the average of the constituent domains. The interactions between the larger domains can in turn be analyzed to determine the large scale behavior of the system, e.g. whether it is in the ferromagnetic or paramagnetic regime. These ideas can be extended by using information theory to analyze the probability distribution of the space of possible dynamics to determine which parameters are relevant at each scale of interest (Bar-Yam 1997) (Zurek 1990). The result is a methodology for producing multi-scale descriptions that ensures that first generation models, consisting of multiple formalisms at different levels of detail, include all and only the relevant parameters at the scale of interest.

Multi-scale descriptions can help us in the process of identifying emergent dysfunctions. Consider the following simple example:

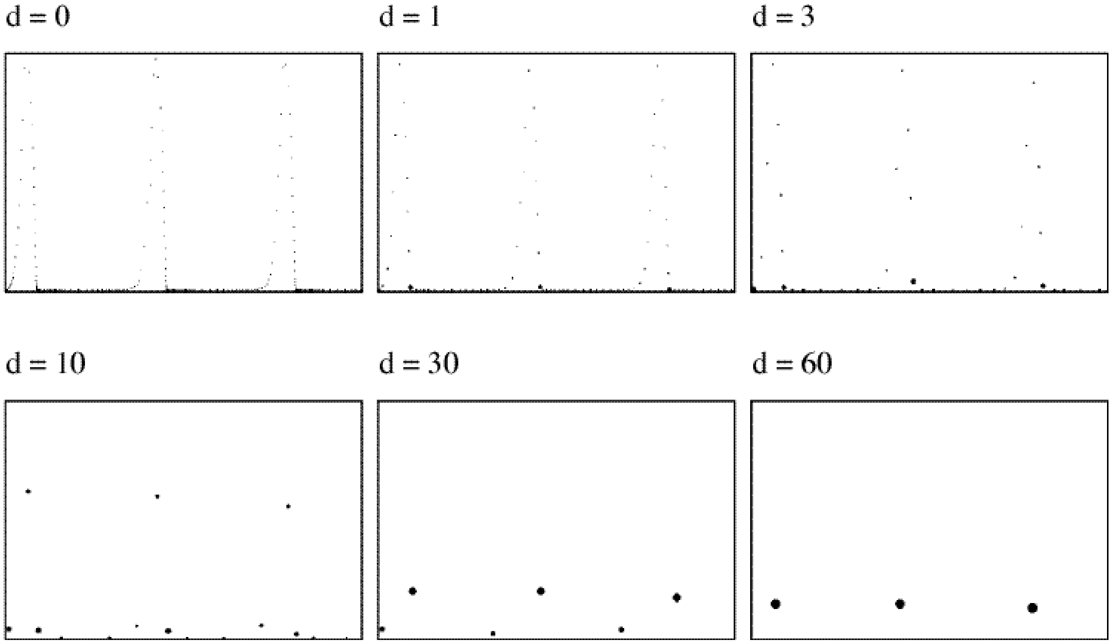

Figure 1: Using Multi-scale Descriptions to Abstract Dynamical Behavior.

This figure shows the clustering of a dynamical equation. Six different scales of descriptions were generated using a progressively higher value of uncertainty in observation (specified by d). The representation of the system progressively simplifies to the point where it captures only the clock-like nature of the system. We can use this information-hiding property of multi-scale descriptions to abstract away less relevant details and reveal larger-scale patterns such as emergent dysfunctions in our peer-to-peer system simulations.

Perturbation Analysis: These peer-to-peer system models can be subjected to *perturbation analysis* (also known as sensitivity analysis) to help us determine what emergent dysfunctions arise as key parameters are varied, e.g. as we increase the number of nodes or connections, or introduce erratic behavior in individual nodes (modeling failures or malicious intervention). It has been shown, for example, that an ecosystem (such as a peer-to-peer system with one or more human and machine "species") can become unstable against weak perturbations when its complexity increases beyond a certain value, and have derived conditions of stability as a function of species diversity and interaction strength.

Previous work of this type has made use of linear analysis techniques, however, which are unlikely to be fully adequate to our purposes because complex systems typically include significant regions where non-linear relationships are important. We will extend perturbation analysis into non-linear regimes using a number of strategies. One is to use stochastic techniques such as Monte-Carlo analysis. Another approach, derived from the complex systems research community, is to extend the concept of the Renormalization Group, incorporating principles of information theory, to enable us to study non-linear systems.

Identifying Equivalence Classes: These analysis techniques leave, however, an important question unanswered. Simply determining the emergent dysfunctions that can occur with a given simulation model may or may not be relevant to any actual peer-to-peer system. How can we generalize the results derived from our simulation models so that they can be used for a wide range of real systems? This brings up a second key contribution of our proposed work. Our analysis of distributed systems to date has revealed that they can be categorized into "equivalence" (or "universality") classes, such that all the networks that belong to that class will have a given set of characteristic emergent dysfunctions. One of the critical characteristics is the way the agents in the peer to peer system coordinate with each other. Work in coordination science (Malone and Crowston 1991) suggests that coordination can be viewed essentially as the management of resource flows amongst agents, and has identified three main classes of coordination mechanism

- *flow* mechanisms that manage the flow of resources from a producer agent to a consumer agent. A supply chain is a prototypical example of a flow management mechanism.
- *sharing* mechanisms that manage the sharing of some commonly-used resource amongst several consumer agents. An auction is a prototypical example of a sharing mechanism.

- *fit* mechanisms that manage the consolidation of several resources into a single larger one. Collaborative design is a prototypical example of a fit mechanism, since designs for multiple sub-components have to be fit together to create a single workable design.

Our research to date has shown that each class of coordination mechanism has it's own characteristic set of emergent dysfunctions. Flow mechanisms with delayed supply feedback, for example, are prone to wild inventory oscillations (Sterman 1994). Sharing mechanisms that are demand-based (e.g. first-come first-serve) are prone to such emergent dysfunctions as resource poaching (Chia, Neiman et al. 1998), tragedy of the commons (Hardin 1968) and simultaneous jump thrashing (Youssefmir and Huberman 1997). Fit mechanisms are prone to such emergent problems as fit conflicts (Klein and Lu 1990) and (when there are asymmetric influences) non-convergence (Klein 2001). Our project will build upon these preliminary insights to identify a more exhaustive range of emergent dysfunction types, as well to delineate more precisely under what circumstances (e.g. network topology, resource scarcity, number of agents) these dysfunctions occur.

<u>A Schema for Organizing Knowledge About Emergent Dysfunctions:</u> However we identify the different classes of emergent dysfunctions, we need a systematic way to organize this knowledge so that it can be applied effectively to real-world peer-to-peer systems. We have developed a simple but effective schema for this, extending concepts developed by the MIT Process Handbook project (Malone, Crowston et al. 1999) (Klein and Dellarocas 2000). The scheme is based on three interlinked taxonomies capturing coordination protocols (AKA mechanisms), their characteristic emergent dysfunctions, and the handlers that can deal with them:

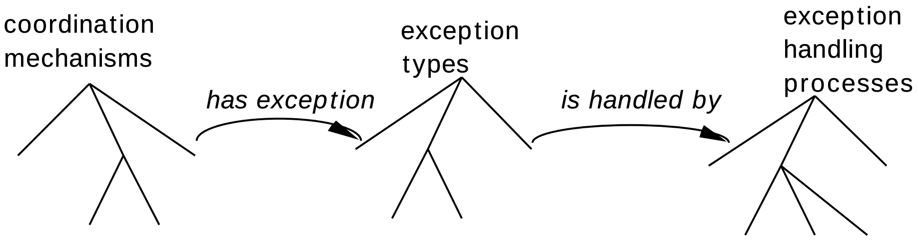

Figure 2. Overview of the schema for emergent dysfunction handling expertise.

The first taxonomy captures peer-to-peer coordination protocols, arranged in an abstraction hierarchy with abstract protocol classes on the left and more specific ones on the right:

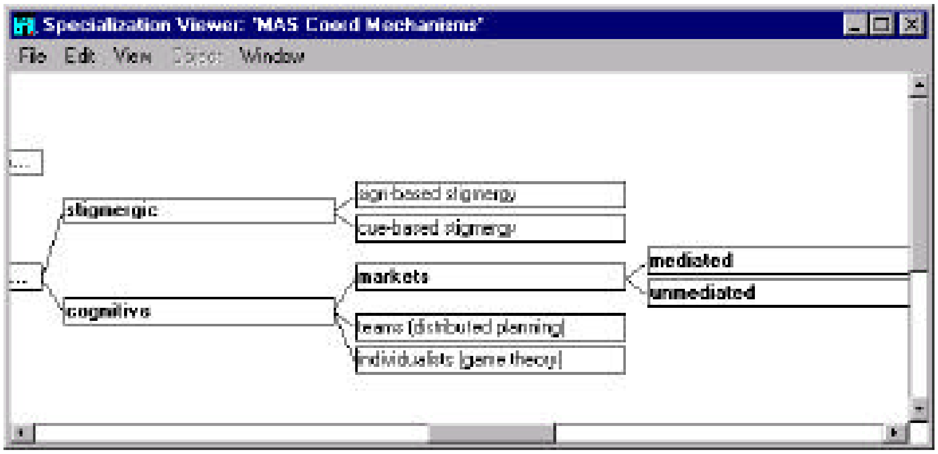

Figure 3. A part of the coordination mechanism taxonomy.

Each protocol has pointers to the emergent dysfunctions that characterize it, themselves stored in an emergent dysfunction taxonomy:

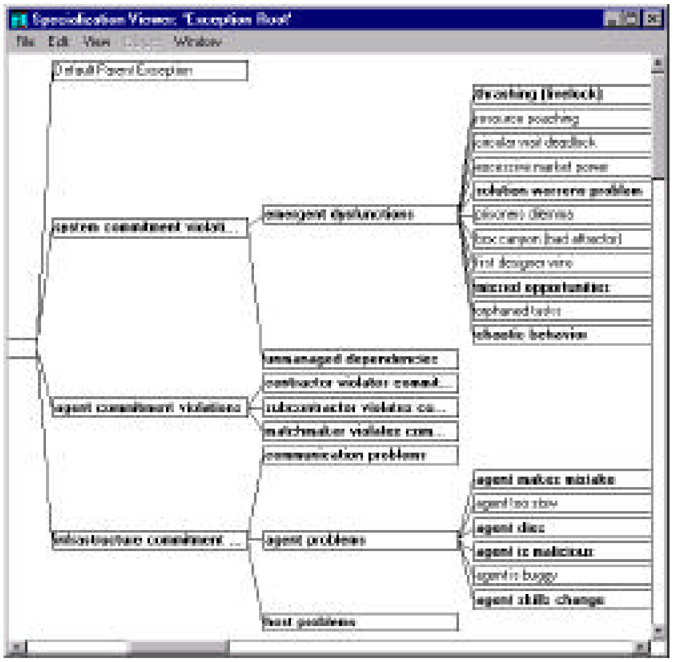

Figure 4. A subset of the emergent dysfunction type taxonomy.

Our work to date, based largely on the review of relevant literatures from computer science and other disciplines, has identified about ## different classes of emergent dysfunctions.

Emergent dysfunctions are themselves linked, finally, to the potential applicable handlers in an emergent dysfunction handler taxonomy. We have found that there are four main types of handlers; those suitable for *anticipating* and *avoiding* emergent dysfunctions (before they occur), or *detecting* and *resolving* them (after they occur).

The power of this approach is that we can use inheritance to simplify identifying the emergent dysfunctions and handlers for a given peer-to-peer protocol. Some emergent dysfunctions are characteristic of a whole class of protocols, and therefore are potentially inherited by all of its subclasses. Any protocol, for example, that implements ‘pull-based resource sharing’ (where resources are allocated by selecting among requests) potentially faces the problem of resource poaching. Similarly, a handler suitable for an abstract emergent dysfunction type is likely to be suitable for subclasses of that type. Preemptive re-allocation of resources, for example, is a reasonable potential candidate for any instance of the resource poaching dysfunction. Finally, if a handler is suitable for an emergent dysfunction, it is likely that subclasses of that handler will be useful for that dysfunction.

2.2. Run-time services for handling emergent dysfunctions

While avoidance is generally better than remediation, it can be difficult or impossible to avoid some important classes of emergent dysfunctions up-front. One reason is that in open peer-to-peer systems, the design of the system is not under centralized control and we have no guarantee that all components will behave in a cooperative way. In other cases (e.g., as in resource poaching, where scarce resources are tied up by lower-priority tasks (Chia, Neiman et al. 1998)), avoiding dysfunctions is theoretically possible but requires inefficient protocols. In such cases, it makes more sense to develop run-time services that monitor the peer-to-peer system for emergent problems and intervene, when necessary, to resolve them. Our team will develop such approaches, building on substantive previous efforts in this area (Klein, Rodriguez-Aguilar et al. 2001). We have already seen that it is possible to identify highly reusable domain-independent emergent dysfunction handling expertise that describes the characteristic failure modes for different peer-to-peer coordination protocols, as well as how they can be handled. Our vision is that components, which we can call ‘sentinels’, will be able to use this expertise to automatically determine, at run time, what emergent dysfunctions can occur, as well as what detection and resolution strategies are appropriate, for a dynamically changing system environment. Our approach instantiates the ideas described using the following functional architecture:

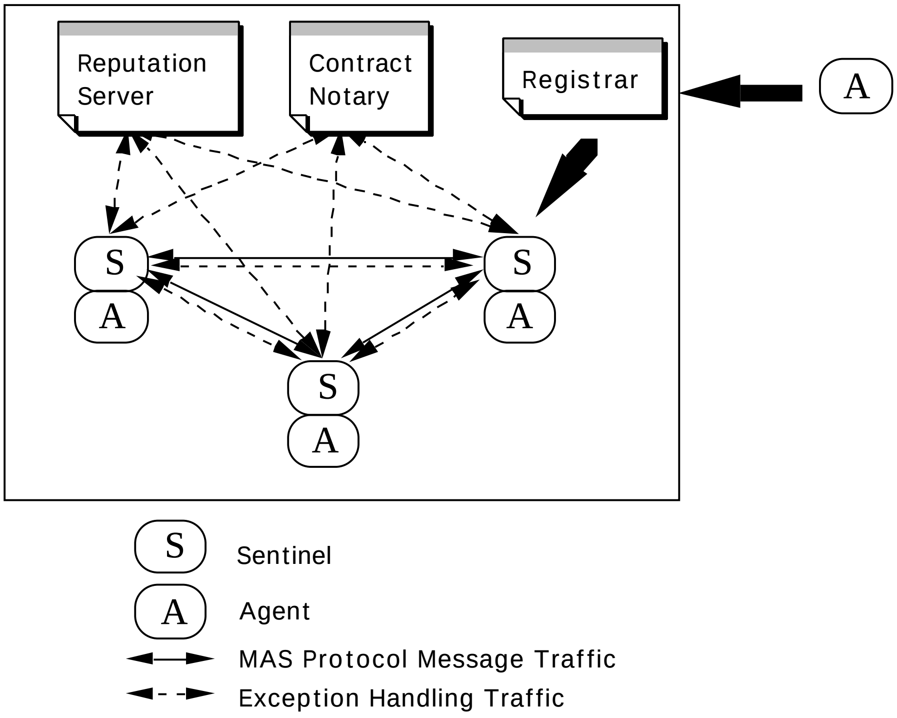

Figure 5. Functional architecture for emergent dysfunction handling services.

When a component joins an open peer-to-peer system served by the emergent dysfunction handling (EDH) services, it must register with the registrar responsible for assigning it a sentinel that will mediate all of the components' further interactions with other components in the system. Sentinels are a central element in this approach. Their role is to observe and influence component behavior as necessary to ensure the robust functioning of the system as a whole. Each sentinel includes a repository of domain-independent EDH expertise that describes the characteristic emergent dysfunctions and associated handlers for the protocol(s) enacted by the components in peer-to-peer system. Sentinels monitor message traffic, use the appropriate detection handlers to uncover potential problems, diagnose the underlying causes to identify the resolution handlers, and enact these handlers to help or remediate the emergent dysfunction. Ancillary services such as the contract notary and reputation server keep track of global state information such as commitment structures and reliability statistics. Components, for their part, must be able to respond appropriately to a relatively small set of EDH directives to support the sentinels.

For defining an EDH services architecture that is both scalable (with respect to number of components) and generic (can be applied to a wide range of peer-to-peer contexts), the architecture is fundamentally distributed: it is based on a distributed sentinel population plus EDH services that are essentially database applications and thus can be replicated using well-known techniques. We have found that, with careful design, it has been possible to minimize EDH related message traffic among these components. The architecture makes few assumptions about the peer-to-peer system the EDH services operate in. Components need implement only a relatively small number of simple EDH directives (e.g. “are you alive?” and “cancel task X”) in order to participate fully. Significant benefit is possible even if the components implement no EDH directives whatsoever. Finally, if sentinels are integrated into the messaging infrastructure, the EDH services can be largely transparent to the components.

This project will take our work further by addressing such challenges as:

- Diagnosing emergent dysfunction causes. As we extend our approach to handle a wider range of emergent dysfunctions, we will increasingly encounter situations where a given emergent dysfunction can have multiple possible causes. How can the EDH services most effectively diagnose (and thereby select interventions for) emergent dysfunctions in that context? Previous work on failure diagnosis can be divided into two broad categories: shallow model and deep model based. The shallow model approach (which includes case-based reasoning as well as heuristic classification) uses experience-based associations to link symptoms with possible diagnoses and (sometimes) possible interventions, and is a good match for partially understood domains such as medicine The efficacy of shallow model diagnosis is a function of the quality and coverage of the knowledge base used, each new artifact requires its own knowledge base (which may become obsolete if the diagnosed entity changes significantly), and there is no guarantee of correct results. The deep model-based diagnosis approach (also known as model-based reasoning), by contrast, uses a complete model of an entity’s behavior to guarantee the correct deduction of possible failure causes from symptoms. The latter approach is not oriented, however, towards diagnosing emergent dysfunctions (e.g. systemic emergent dysfunctions such as thrashing or component commitment violations such as distributed denial of service attacks) wherein dysfunctions result from apparently normal local behavior. We propose to explore integrating elements of both shallow and deep-model based diagnosis in order to address the unique challenges of emergent dysfunction handling in open systems. The commitment graph revealed by RCV analysis of a coordination mechanism provides a causal structure that can be used as a basis for model-based diagnosis of which commitment failed. Model-based reasoning can not be used to diagnose the underlying causes of the commitment violations, however, because open system components are inherently ‘black boxes’ unlikely to have accessible behavioral models. Shallow model approaches can be used at this point to suggest possible emergent dysfunction causes as well as potential resolutions.

- Dealing with components that include some native (‘survivalist’) emergent dysfunction handling capabilities, and may therefore prefer to use their own techniques rather than outsource handling of some emergent dysfunctions to the EDH services? One possibility is to create the notion of an explicit EDH ‘contract’ wherein the components and EDH services specify their agreement, for each of the emergent dysfunctions characteristic of their protocol, concerning which emergent dysfunctions they wish to outsource and which they prefer to handle themselves.

3. Work Plan

Task 1: *Acquire Emergent Dysfunction Handling (EH) Expertise*: The goal of this task is to systematically perform an emergent dysfunction sensitivity analysis for all the most important protocols. Our initial focus will be on market mechanisms. Such mechanisms are unique in being scalable to large numbers of components, are backed up with substantive theoretical and empirical results from economics, computer science and other disciplines, and are immediately applicable to many of today’s most compelling practical challenges such as e-commerce.

Task 2: *Develop Emergent Dysfunction Handling Services*: We will define the component development guidelines and develop the EH service components needed to exploit the EH expertise acquired above in order to improve the robustness of open peer-to-peer systems, in a way that is both scalable and generic.

Task 3: *Evaluate Services in Simulated peer-to-peer System Environment*: Our work will evaluate whether our domain-independent EH services approach effectively increases robustness in open peer-to-peer system contexts. We will evaluate the EH services in a range of realistic simulated peer-to-peer scenarios, designed based on the analysis of the abstract properties of representative problem domains such as information retrieval and e-commerce.

Task 4: *Disseminate Results*: The results of our work will be disseminated via publications as well as planned workshops and special issues in this area.

4. Summary

Peer-to-peer systems, as we have seen, are emerging for fundamental reasons as the dominant form of large networked software system for both commercial and military contexts. The primary outcomes of this project will be to increase the reliability of such systems by greatly reducing the incidence of potentially debilitating emergent dysfunctions. In particular, we will create:

- ♦ A systematic delineation of the kinds of emergent dysfunctions that can arise in networked peer-to-peer systems, organized by the abstract network characteristics (e.g. topology, resource sharing protocol) that make these dysfunctions likely. These will be identified making use of multi-scale models that incorporate dynamical models of user demand far more sophisticated than the simple static averages that have been used to date in network evaluations.

- ♦ A set of validated design principles for how to create peer-to-peer systems that avoid many of these emergent dysfunction types.
- ♦ The design for run-time services appropriate for addressing emergent dysfunctions which can not be effectively avoided using up-front design decisions.

More generally, this work will lead to the development of methodological refinements in modeling complex systems, using multi-formalism multi-scale models, that will be applicable to understanding a wide range of pressing complex systems problems. Environmental degradation, and many kinds of social unrest including those that manifest as terrorism, for example, can be viewed as emergent dysfunctions in complex systems.

5. References

Bar-Yam, Y. (1997). Dynamics of complex systems. Reading, Mass., Addison-Wesley.

Binder, P. M. and J. A. Plazas (2001). “Multiscale Analysis of Complex Systems.” Physics Review E **63**.

Chia, M. H., D. E. Neiman, et al. (1998). Poaching and distraction in asynchronous agent activities. Proceedings of the Third International Conference on Multi-Agent Systems, Paris, France.

Hardin, G. (1968). “The Tragedy of the Commons.” Science **162**: 1243 - 1248.

Klein, M. (2001). What Complex Systems Research Can Teach Us About Collaborative Design. International Workshop on CSCW in Design, London. Ontario, Canada, IEEE Press.

Klein, M. and A. Baskin (1990). A Computational Model for Conflict Resolution in Cooperative Design Systems.

Klein, M. and C. Dellarocas (2000). Towards a Systematic Repository of Knowledge about Managing Multi-Agent System Exceptions. Cambridge MA USA, Massachusetts Institute of Technology.

Klein, M. and S. C.-Y. Lu (1990). “Conflict Resolution in Cooperative Design.” International Journal for Artificial Intelligence in Engineering **4**(4): 168-180.

Klein, M., J. A. Rodriguez-Aguilar, et al. (2001). “Using Domain-Independent Exception Handling Services to Enable Robust Open Multi-Agent Systems: The Case of Agent Death.” Autonomous Agents and Multi-Agent Systems.

Laboratory, L. A. N. (2001). TRANSIMS Reports.

Malone, T. and K. Crowston (1991). Towards an Interdisciplinary Theory of Coordination, MIT Center for Coordination Science.

Malone, T. W., K. Crowston, et al. (1999). “Tools for inventing organizations: Toward a handbook of organizational processes.” Management Science **45**(3): 425-443.

Murata, T. (1989). “Petri Nets: Properties, Analysis and Applications.” Proceedings of the IEEE **77**(4): 541-580.

Sterman, J. D. (1994). Learning in and about complex systems. Cambridge, Mass., Alfred P. Sloan School of Management, Massachusetts Institute of Technology.

Waldrop, M. (1987). “Computers amplify Black Monday.” Science **238**: 602-604.

Youssefmir, M. and B. Huberman (1995). Resource contention in multi-agent systems. First International Conference on Multi-Agent Systems (ICMAS-95), San Francisco, CA, USA, AAAI Press.

Youssefmir, M. and B. A. Huberman (1997). “Clustered volatility in multiagent dynamics.” Journal of Economic Behavior & Organization **32**(1): 101-18.

Zurek, W. H., Ed. (1990). Complexity, Entropy and the Physics of Information. Reading, MA, Addison-Wesley.