A. Arpaci-Dusseau **Department of Computer Science**
CS739: Distributed Systems **University of Wisconsin, Madison**

Disconnected Operation in the Coda File System (SOSP'91)

1 Motivation

- What were the goals of Coda?
- What assumptions did Coda make?
- How good of a job did the designers do of predicting technology trends?
- Coda developed from AFS. Briefly, how did AFS work with regard to caching files?? What type of data consistency does AFS provide?
- Replication is often used to increase availability, but there are trade-offs that must be considered. Is it possible to simultaneously achieve perfect **consistency** and **availability** when suffering from network partitions? Why or why not? Which does Coda place more emphasis on?
- When a network is partitioned, replicas can be controlled with either pessimistic or optimistic replica control. What is *pessimistic* replica control? What are the pros and cons of it? Why don't leases solve the problem?
- What is *optimistic* replica control? What are its pros and cons? Why was optimistic replica control chosen in Coda? Can you think of an environment where pessimistic replica control would be more appropriate?
- Coda performs replication on both the servers (VSG, volume storage group) and clients. What are the differences between these two types of replicas? What does Coda do if some, but not all, servers are available? With a different view of servers, how might you design a file system for disconnected operation?

2 Detailed Design and Implementation

- Clients are managed by a software layer called Venus. How does the state and behavior of Venus change as the client becomes disconnected or connected?
- Consider the hoarding state first, in which Venus attempts to hoard useful data in anticipation of disconnection. The challenge for hoarding is that the amount of cache space on the clients is, of course, limited. During hoarding, what tensions must Venus balance in how it manages the client cache? How does Venus decide what is cached? (What information is given infinite priority in the local cache? Why?)

- Is Venus during the hoarding stage identical to AFS? Why might the performance of Coda Hoarding be worse than AFS?
- Imagine that Venus includes a command so a user can specify that disconnection is about to take place. How should Venus respond?
- During emulation, Venus on the client performs many of the actions normally handled by the servers. What types of tasks does this include? How does Venus record enough information to update the servers during reintegration? How does Venus save space? What happens when all space is consumed?
- During reintegration, Venus propagates changes made during emulation to the servers and updates its cache to reflect current server state. What are the steps of reintegration? Under what circumstances will the replay fail? How is failure detected? What happens when the replay fails? Do you think Coda chose the right level of granularity for conflict resolution?

3 Status and Evaluation

- In their Evaluation, they look at 3 questions. How long does reintegration take? How large a local disk does one need? And, how likely are conflicts? Which question do you think has the most impact on whether or not Coda could be successful?
- Question 1: About how long is reintegration expected to take? Why is the time for this step crucial? How are technology trends likely to impact this time? Is a design change needed?
- How did they determine the size of a needed local disk? How are technology trends likely to impact this? Is a design change needed?
- How likely is a conflict during reintegration? Will technology trends impact this? Is a design change needed?

4 Conclusions

- Coda handles both voluntary and involuntary disconnection of a client from the network. Where could Coda have made different (or simpler) decisions if they had handled only voluntary disconnection?
- If you were designing a file system for disconnected operation today, what would you do differently?