

Peer-to-peer brokering of planning meta-data

Johannes Oudenstad,
Frank Eliassen,
Eli Gjørven,
Romain Rouvoy

Agenda



- QuA planning middleware
- Motivations
- Design of P2P based broker
- Mapping mirrors to P2P keys
- Implementation & experimentations
- Conclusion & perspectives



QuA planning middleware

- Service mirrors [bracha04]
 - *Mirror-based service reflection*
 - *Obtained from a pluggable broker component*
- Planning based adaptation [eliassen06]
 - *Applications specified by behavior (service types)*
 - *Service planner component*
 - Configures and reconfigures applications
 - Uses broker component to find alternative service mirrors
- QuA capsule
 - *Represents the local runtime environment of a specific QuA platform*
 - *Hosts repositories for service blueprints*
 - *Service mirrors for each capsule are advertised to the broker*

Motivations



- P2P architectures become more usual in distributed applications
- A component that gives the opportunity of sharing service offer space between nodes in an easily configurable manner is interesting.
- By using peer-to-peer technology, it would be possible to build a trading service that compared to current solutions is
 - *More **scalable***
 - *More **resistant** to nodes joining and leaving (even unexpectedly)*
 - *More **available** and more easily **configurable***



Design of P2P based broker

- Use a structured P2P overlay to distribute service mirrors
- Map service mirrors to nodes in a way that ensures a fairly even distribution
- Technology independent design (but requires a DHT-like solution)
- Nodes working together to implement a distributed broker service
- P2P-broker consists of 3 parts
 - *QuA part*: Contains broker logic
 - *Glue part*: Glues the upper and lower parts together. Assists in replication
 - *Overlay part*: Responsible for communication between nodes

Mapping service mirrors to P2P keys

- How to build overlay-specific keys from service mirrors?
- Intuitively: *Use type as key*
 - *Some types are more often used than others*
 - Problem: All queries for a given type from all QuA nodes will end up at one unlucky node
 - Problem: Quickly a lot of mirrors to filter through for “popular” QuA types
- Alternative: *Involve meta-information in key building*
 - *We use static properties with enumerated value range*
 - *Trading increased storage cost (but more evenly distributed storage) for lower search times*
 - *2 advantages:*
 - **Smaller search spaces**
 - **Increased parallelism**

Implementation & experimentations

- We have a working solution
 - *Uses the Java QuA and Java FreePastry implementations*
 - *Initial tests are performed with multiple nodes running on one computer*
- Scalability/distribution
 - *Assumption that a good distribution on nodes ensures a scalable solution*
 - Pastry team has shown that the Pastry mechanisms are scalable
 - *Service mirrors are fairly evenly distributed in experiments with several hundred nodes and tens of thousands of service mirrors*
 - *But: Effects of real-network latencies on service planning not clear*
- Availability / self-organization
 - *Replication keeps service metadata highly available*
 - *When nodes join or leave, the service mirror requests are immediately transferred to the right responsible node*
- Further work: *More testing in a real distributed environment is needed*



Conclusion & perspectives

- Planning-based adaptation middleware can benefit from P2P metadata discovery in P2P environments
- In particular, our P2P broker supports
 - **Node failures** through replication
 - **Larger offer space** through resource sharing
 - **Node dynamicity** that can join / leave the network as they like
 - **Acceptable response time** through resource distribution
- Perspective
 - Use P2P to distribute service blueprints
 - Use P2P to distribute service planning

Questions & references

- G. Bracha and D. Ungar. **Mirrors: Design Principles for Meta-level Facilities of Object-Oriented Programming Languages.** In *19th Annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA)*, pages 331–344, Vancouver, BC, Canada, 2004. ACM Press.
- F. Eliassen, E. Gjørven, V. S. W. Eide, and J. A. Michaelsen. **Evolving Self-Adaptive Services using Planning-Based Reflective Middleware.** In *5th International Middleware Workshop on Adaptive and Reflective Middleware (ARM)*, volume 190 of AICPS, page 6. ACM, 2006.
- J. Oudenstad. **Design and evaluation of a QuA implementation broker based on P2P-technology.** *Master thesis. Supervisor: Frank Eliassen. University of Oslo, 2007.*
 - <http://www.duo.uio.no/publ/informatikk/2007/52908/Oudenstad.pdf>