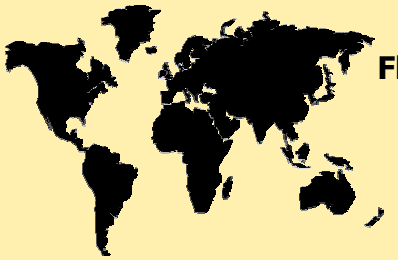


ServiceGlobe: Flexible and Reliable Web Services on the Internet*

Markus Keidl, Stefan Seltzsam, and Alfons Kemper
Universität Passau, Germany
http://www.db.fmi.uni-passau.de/projects/sg/

*This research is done in cooperation with the Advanced Infrastructure Program (AIP) group of SAP.

1 The ServiceGlobe System

- ServiceGlobe is an open, distributed, and extensible service platform developed for research purposes
- Fully implemented in Java 2
- Based on standards like XML, SOAP, UDDI, WSDL,…
- Offers standard functionality of a service platform like secure communication, a transaction system and a security system
- Supports mobile code, i.e., services can be distributed and instantiated during runtime on demand

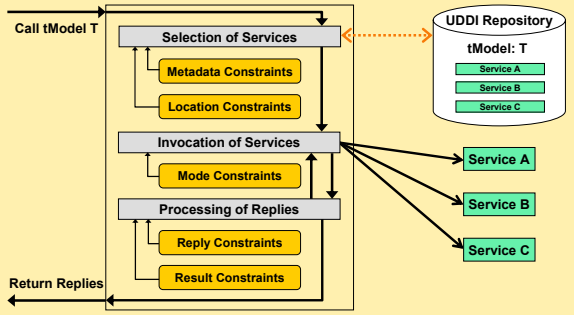
2 Dynamic Service Selection

- Select and invoke Web services at runtime based on a technical specification
  ⇒ Abstraction from actual services
- Service platform selects suitable Web services utilizing UDDI and tModels
- tModel = Classification of a service's functionality and a formal description of its interfaces
- Constraints to control selection of services, their invocation, and processing of replies

3 Constraints

- Metadata Constraints: Select services based on their metadata (UDDI and/or other metadata repositories)
- Location Constraints: Select services based on their location (stored in UDDI)
- Mode Constraints: Define the number of Web services to invoke when calling a tModel (one/some/all)
- Reply Constraints: Filter replies based on their content and their properties
- Result Constraints: Control termination of dynamic service selection, e.g., specify a timeout

4 Execution of Dynamic Service Selection

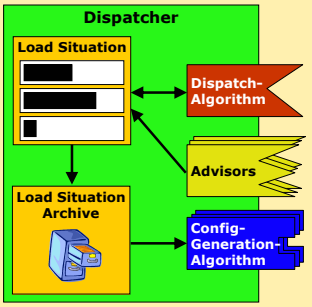


5 Generic Dispatcher Service

- A single service instance is not sufficient to provide low response times and high availability.
- Therefore, several instances of a service run concurrently on several hosts and our dispatcher avoids load skews performing load balancing.
- This dispatcher service (Layer-7 switch), acts as proxy for arbitrary services and is a generic solution for such situations.
- The dispatcher is implemented as a regular service. Therefore, it is more flexible, extensible, and seamlessly integrated into the service platform.
- Special feature: automatic service replication

6 Modular Architecture of the Dispatcher

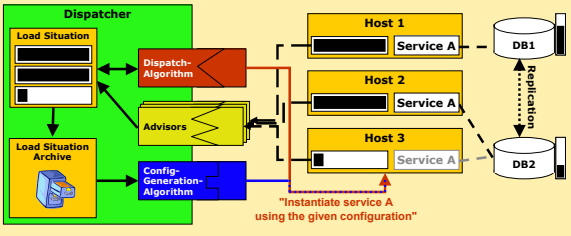
- The dispatch module implements the actual dispatching strategy.
- Advisor modules are used to collect data about the load situation.
- Config modules are used to generate config documents for new service instances (see automatic service replication).



7 Automatic Service Replication

- If all available service instances are running on heavily loaded hosts and there are hosts available having a low workload, the dispatcher can decide to generate a new service instance using automatic service replication.
- Using the appropriate config module for a service, the dispatcher can generate a configuration document for the new service instance, e.g., to determine which database instance of a replicated database system to use.

8 Example for Automatic Service Replication



All instances of service A are running on heavily loaded hosts. Therefore the dispatcher decides to generate a new Instance of service A on host 3. The config module for service A uses the load situation archive and determines that DB2 had a lower average load in the past, so the new instance of service A should use the DB2 instance of the replicated database system.