d

Mediator system for robotic applications

Rubanraj Ravichandran

Master Thesis Proposal
Master of Autonomous Systems

Department of Computer Science
University of Applied Sciences Bonn-Rhein-Sieg

Supervisors:
Prof. Dr. Erwin Prassler
Advisors:
Nico Huebel
Sebastian Blumenthal

November 12, 2018

1 Introduction

Robots generate large amount of data from different types of sensors attached to it and also from its hardware components. In our previous research work [**?**], we have conducted an extensive qualitative and quantitative analysis to find better databases and architectures that effectively store these data and consume it for further operations. Results from our previous work shows that, a single database is not suitable for every robotic scenarios. For example, in terms handling large BLOB data, mongoDB stored them faster but reading the data was slower compared to couchdb [**?**]. Also, to complete a given task robots depends on multiple source of information from internal sensors, as well as external sources for example world model, kinematic model, etc..

Adoption of multiple databases for robotic applications requires unique way of mediation to view multiple databases as a single federated database. Mediator approach helps to integrate data from different sources and produce a single result back to robots. Mediator abstracts the information of how data is being stored in different data sources from robot, and allows robotic applications stream data to mediator independent of databases used in the back-end.

To Map the data generated by robots with multiple databases, mediator system requires a proper data model predefined in the context of robotic applications. Modeling robot generated data helps to generalize the structure of data and defining relations between different objects in the robot systems. If we have a well defined robotic data models, then mediator will get the ability to mutate or query data from different data sources. Also it is important that, these data models should be extended to any robotic use-cases.

As mentioned in these papers [**?**],[2],[3], mediators are being used to integrate data from different data sources and few architectures supports single data model (e.g SQL), and others supports for different data models (e.g SQL,NoSQL, document store, etc..). Also, they are differ from query languages, ease of implementation, components used in their architecture. This project focuses on defining suitable data models for robotic applications

and implementing a mediator system which act as a middle-ware between robots and databases.

2 State of the art analysis

?] proposed an active mediator architecture to gather information from different knowledge base and combine them to a single response. AMOS[1] architecture uses Object-Oriented approach to define declarative queries. This distributed architecture involves multiple mediator modules to work collaboratively to collect the required piece of information and produce final result. Primary components of AMOS architecture are,

- Integrator - Gather data from multiple data sources that have different data representations.
- Monitor - Monitor service always watch for any data changes and notifies the mediators. This is helpful in the case where system needs an active updates to change its current task.
- Domain models represents the models related to application which helps to access data easier from any database through a query language.
- Locators helps to locate mediators in the network.

Integrator module is built with two internal components called IAMOS[2] and TAMOS[3]. First Integration AMOS parse the query and send individual requests to Translational AMOS modules which are responsible for heterogeneous data source. Then,all TAMOS modules return the individual results to IAMOS for integrating all the results. To query multi databases from IAMOS, IAMOS servers are mapped with TAMOS servers w+ith the help of Object-Oriented query language.

[1] Active Mediators Object System
[2] Integration Active Mediators Object System
[3] Translation Active Mediators Object System

?] developed a heterogeneous multi-database system called Pegasus that supports multiple heterogeneous database systems with various data bases models, query languages and services. Pegasus predefines its domain data models based on object oriented approach and also supports programming capabilities. These objects are created and mapped with the types and functions with the help of HOSQL[4] statements. HOSQL is a declarative object oriented query language which is used by Pegasus to manipulate data from multiple data sources.

Pegasus system supports two types of data sources, local and native data sources. Whenever a new data source joins Pegasus system, schema integrator module imports schema from data source and update its root schema with the new schema types. The final integrated schema shows the complete blueprint of the different data sources participates in the data integration. Pegasus system work-flow is comparatively similar to AMOS architecture, but they use different query language and data modeling strategies.

?] developed project Tsimmiss extract information from any kind of data source and translates them to a meaningful common object. Unlike AMOS and Pegasus, Tsimmiss follows a straight forward approach to define the data model which is a self-describing object model. Each object must contain a label, type and value itself. Label can be used by the system to understand the meaning of the value and type shows the observed value type. Objects can be nested together to form a set of objects.

Tsimmiss tool offers a unique query language called OEM-QL and this language follows the SQL query language pattern to fetch the data from mediators. Mediators resolves the query and send separate requests to respective data sources to retrieve the information and merge them together to give a single response back to user. During data integration process, Tsimmiss removes possible duplicates to avoid redundancy in the response. Also, Tsimmiss bundles a default browser tool to query data using OEM-QL language.

[4]Heterogeneous Object Structured Query Language

In the articles discussed above, mediators are targeted to extract information from different data sources that could be different databases or data from file-system. But Rufus system proposed by **?**] focus only on semi structured data stored in file system for example documents, objects, programming files, mail, binary files, images etc. Rufus system classifier automatically classifies the type of file and apply a scanning mechanism on those files to extract the required information and transform them to the appropriate data model which is understandable by Rufus system. Rufus can classify 34 different classes of files. In terms of query language, Rufus can apply simple object predicates and finding text from the extracted information from the documents or files.

?] proposes a Mediator Specification Language that helps the mediator to understand the schema and integrates the data from unstructured or semistructured source. MSL overcomes the major problems in existing mediator systems for example,

- Schema domain mismatch
- Schematic discrepancy
- Schematic evoluation
- Structure irregularities

During translation of original information from different sources to a single object it should be important that, all data sources should have the required attribute and the name of the attribute should be same. Otherwise, mediator system will not be able to process the information to a single answer. External predicates and Creation of the Virtual Objects in MSL solves the problems mentioned above.

?] built a mediator which is flexible to map domain level query different datasources and efficient to plan the query execution to reduce the overall execution time. Information source models provides relations between the super class and subclasses, and also the mapping between the domain models and information from heterogeneous sources. SIMS uses Loom as a representational language to make objects and relationship between them. SIMS

supports parallel query access plan that makes the mediator to access information independent of data sources and the user will get the final answer as quick as possible.

Many mediator systems developed in the past to support integrate heterogeneous information from different data sources. All of them built with different architectures, query language, and execution optimization. In our mediator approach we focus mainly on,

- How different type of robot generated data will be stored in multiple data sources?
- A unique data model to represent the components attached with each robot and data generated by them.
- Semantic query language to communicate with mediator. Unlike traditional query languages we would like to attempt new way of querying data for example Graphql.
- GUI tool to visualize and analyze the robot generated data in a meaningful way.

3 Problem Formulation

Our previous work results reveals not all databases reacts in a similar fashion for different heterogeneous data from robotic applications. Also, there are no data models has been defined in the context of robotic application sensor data. For example, black box designed for ROPOD project simulation uses mongoDB to store raw sensor data with out any data model.

(contact alex or santhosh to see the current implementation and complete the problem here)

In terms of supporting multiple databases setup for robots, we need a systematic approach to store and retrieve data from external sources and, a well defined data model that can map components and sensors to a robot and also with the world model.

These are the major problems will be discussed and addressed in this work,

- Data model for robotic applications
- Implementation of single mediation layer to connect between robots and databases (different data model)
- Unique query language that can be used by robots to talk with different databases via mediator.

3.1 Use cases

Need to be defined

4 Approach

To find the best data models for robotic applications and build a scalable mediator, we begin with SOA analysis to find out approaches that has been followed through before for similar data integration applications. After defining the data model, list of recent querying techniques will be collected and reviewed based on the features and possibility of adopting them with the mediator as a base. In the review we would like to consider current well known querying techniques such as Graphql, something 2, something 3. At first, our mediator will support only the databases which are selected based on the results from our previous research work. Then, schema's will be defined to map the data being generated by the robot and the data sources. To reduce the complexity of identifying appropriate datasources by the robot, in our architecture mediator will dynamically choose the datasource respective to the type of data that robots want to store and retrive. Still the configuration will be adjustable according to the scenarios. Finally to visualize the integrated data from mediator in a meaningful way, a GUI application will be developed based on Node.js stack.

5 Workplan

This workplan shows the major decomposition of the workpackages, start time and end time. This project starts on August 15^{th}, 2018 for 6 months duration and ends on February 15^{th},2018.

5.1 Work packages

6 Deliverables

6.1 Minimum

- Report on state of the art analysis.
- Collection of existing mediator architectures which can be adopted to robotic applications.
- Data modeling in the context of robotic applications.
- Finding unique and semantic query language to communicate with mediator.

6.2 Expected

- Designing and implementing mediator system that supports storing and retrieving robot generated sensor data from different data sources dynamically.
- Suitable data model and relationships for multi robot use-cases.

6.3 Maximum

- Developing a GUI tool to interactively communicate with mediator and visualize the sensor data in a meaning way.