A Framework for Architecture-driven Service Discovery

A. Kozlenkov **V. Fasoulas** **F. sanchez** **G. Spanoudakis**
A. Zisman

Software Engineering Group
Department of Computing
City University
London - UK

Outline

- Motivation
- Framework Overview
- Example
- Query Specification
- Query Execution Engine
- Query Results
- Conclusion and Future Work

Motivation

- Development of *Service Centric Systems (SCS)*: construction of software systems based on the composition of autonomous web-services
- Need to extend software development practices with new processes, methods, and tools to assist the engineering of complex and dependable SCS
- Discovery and composition of services at different stages of the development life-cycle of a system

Architecture-driven Service Discovery Framework

Framework Overview

Definition: Identification of services that can provide the functionality and satisfy quality properties and constraints of SCS as specified by its design models

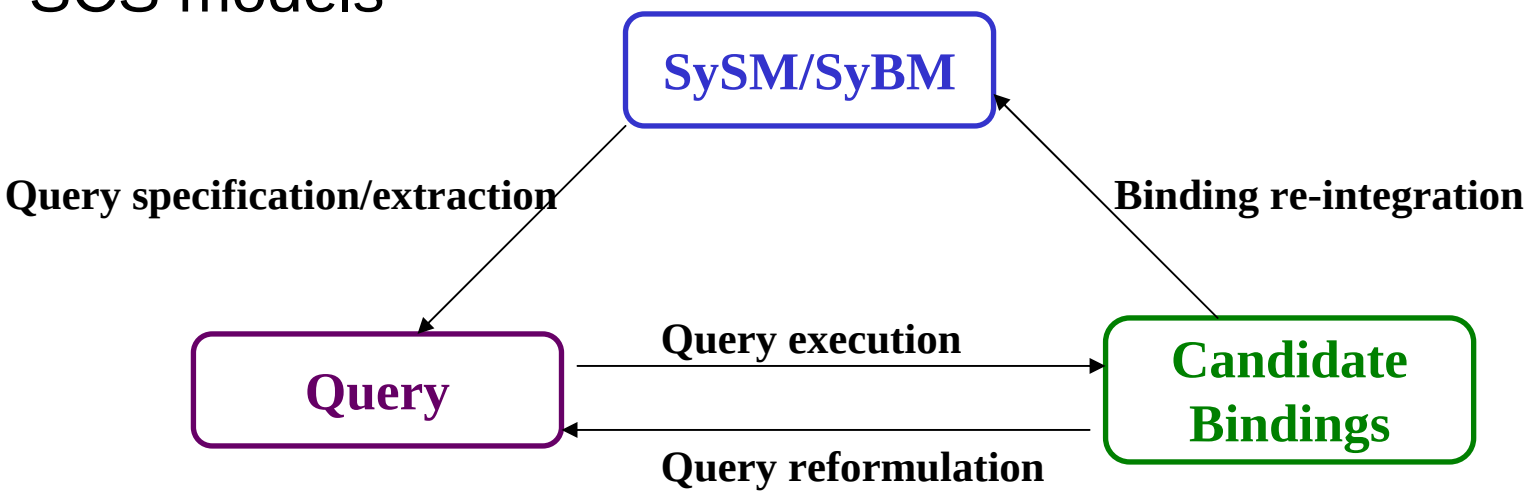
Requirements/Challenges*:

- Extraction of queries from SCS architecture and design models
- Provision of a query language to support combination of prioritised functionality and quality properties
- Efficient matching of queries against service specifications and return of services with varying degrees of match
- Assistance to designers to select discovered services
- Integration of discovered services into an iterative design process (model re-formulation)

*Identified by industrial partners in the EU SeCSE project

ASD Process

- Iterative architecture-driven service discovery process: queries are derived from SCS design models and discovered services are used to amend and re-formulate SCS models



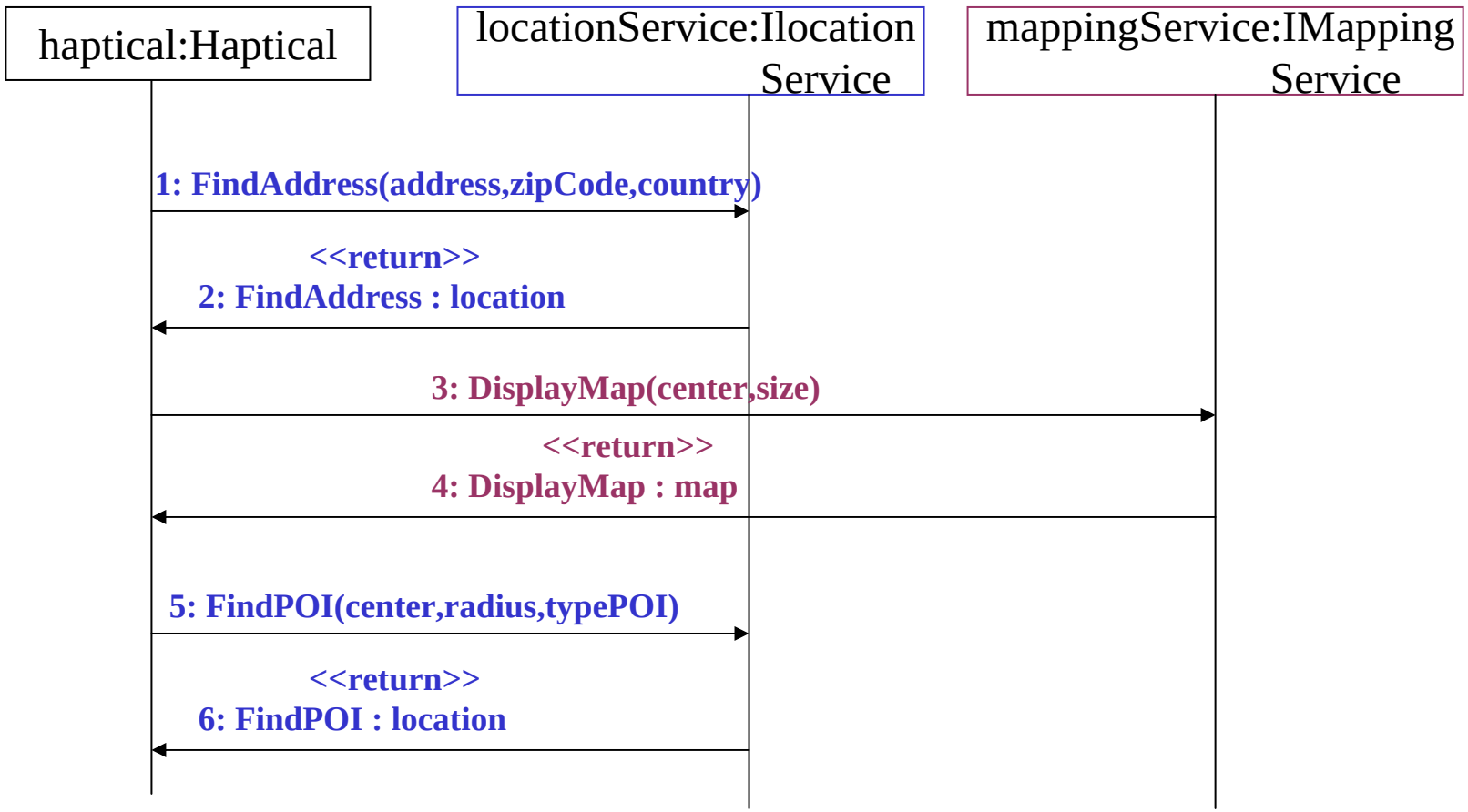
(SyBM) *System Behavioural Model*: describes interactions between operations of SCS (UML sequence diagram)
(SySM) *System Structural Model*: describes the types of the operation parameters and constraints (UML class diagram)

ASD Framework

- UML 2.0 Integration Module
  - Extraction of queries expressing the functionality, properties, and constraints from design models
  - Integration of discovered candidate services into the design models
- Query Execution Engine
  - Search for services in different service registries based on similarities between queries and services (graph-matching algorithm)
  - Service specifications: *facets* describing different aspects of services - interface description (WSDL), behavioural description (BPEL4WS), semantic description (OWL, WSMO), other (cost, quality)

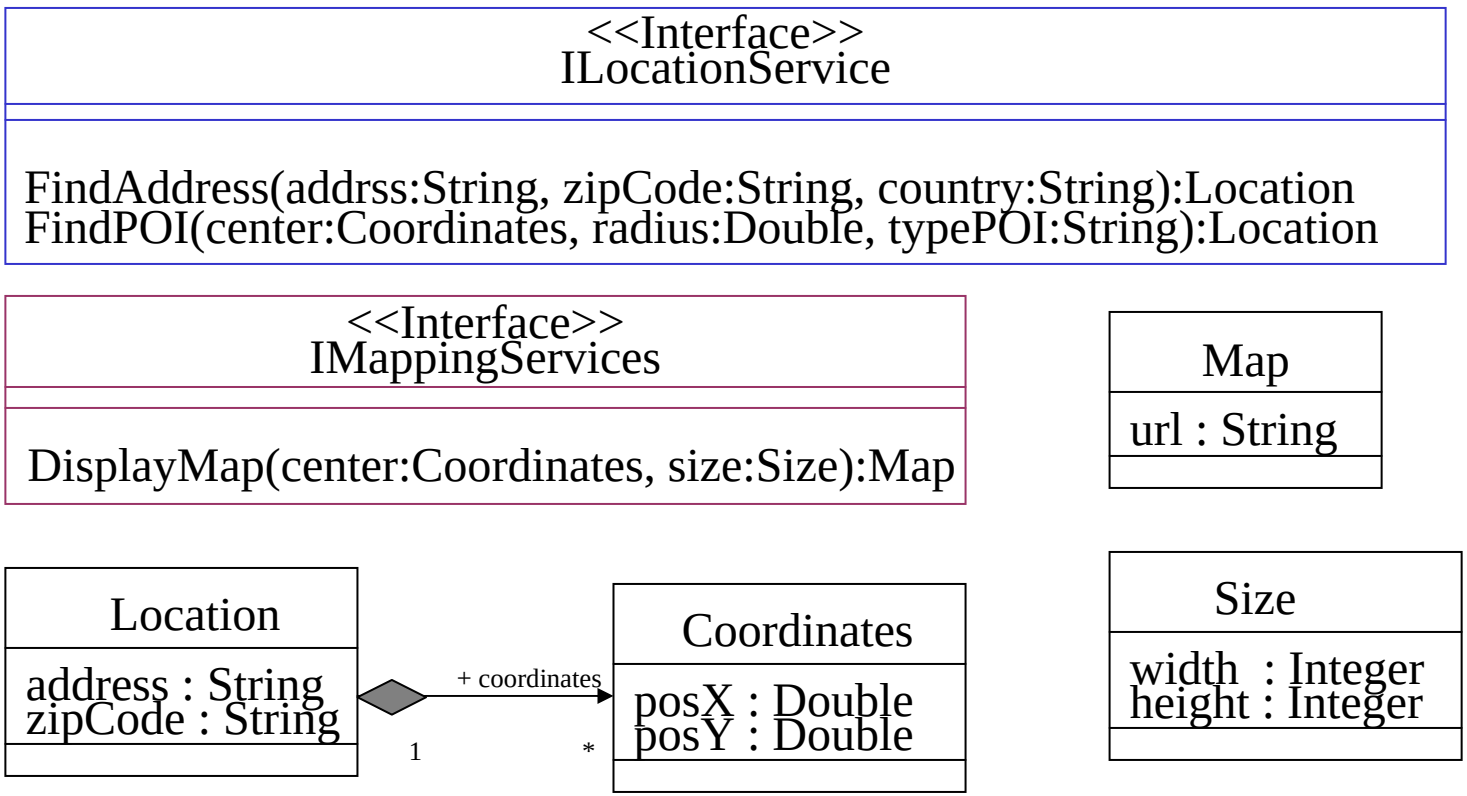
Example (SyBM)

Global Positioning Service Centric System



Example (SySM)

Global Positioning Service Centric System



Query Specification (1/3)

- A query is specified by system designer by selecting messages that should be realised by operations of services to be discovered and specifying constraints

> Query = copy of an interaction from SyBM (I') + selected messages in I' + constraints

- Query and results are specified by *ASD profile* - set of stereotypes for different UML elements found in:
  - Query interaction (messages)
  - Query results (messages, services)
  - SySM referenced by elements of the query interaction (operations, classes) or result parameters

Query Specification (2/3)

- Stereotypes of interaction messages
  - <<query_message>>: service operations to be discovered
  - <<context_message>>: constraints for the query messages
  - <<bound_message>>: bound to concrete operations that have been discovered in previous iterations
- *Query Parameters*: scalar values that limit the search space and amount of information returned by the execution engine (number of services to be returned)
- *Query Constraints*: provide specific selection criteria (condition about services and operations)
  - Type: hard or soft
  - Body: OCL expression

    e.g.: *self.description.Provider.contains('nameProvider')*
  - Weight: optional weight if constraint is soft [0.0 and 1.0]

Query Specification (3/3)

ASD Query Package = M_Q U DC_Q U IC_Q

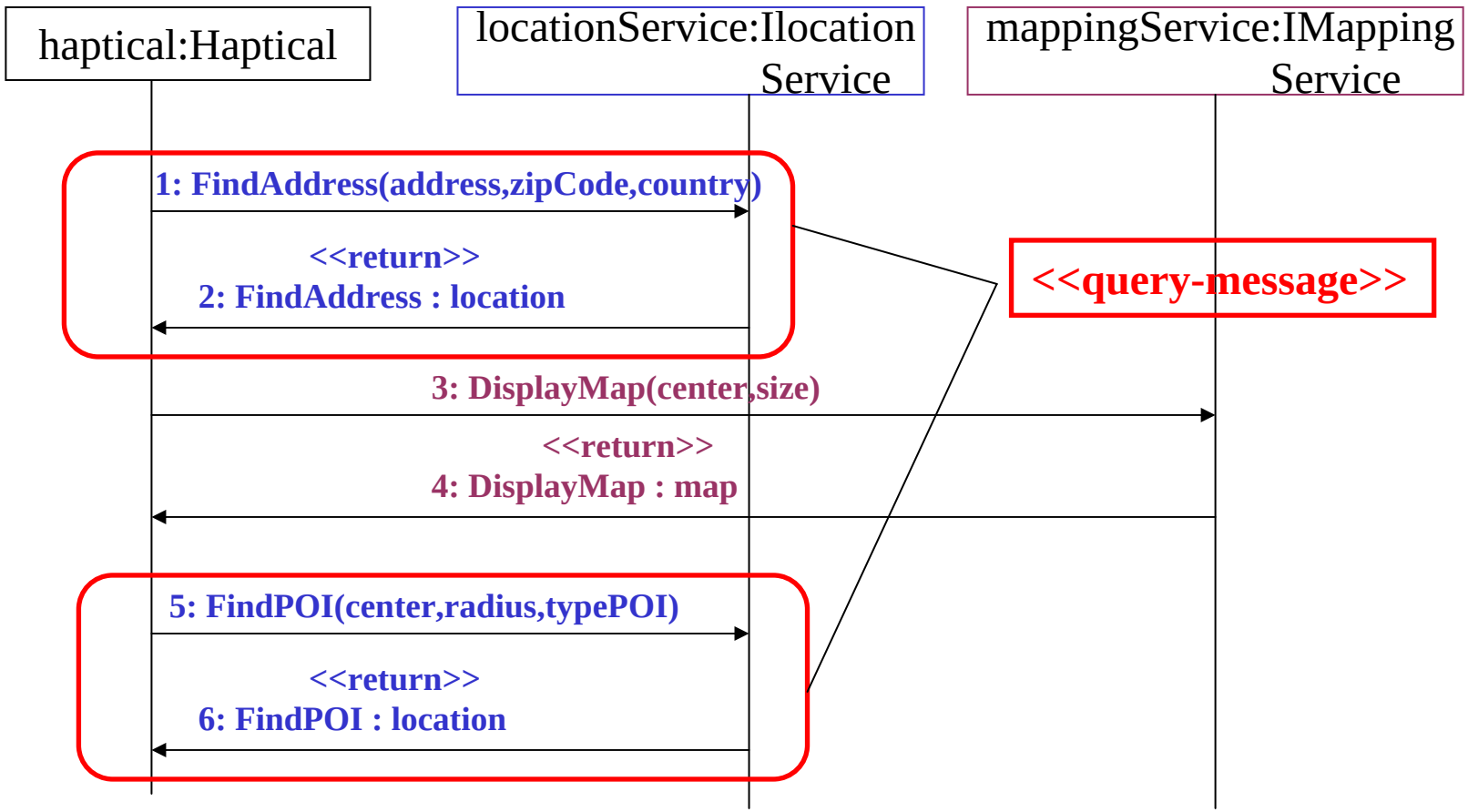
M_Q : set of query and context messages

DC_Q : classes defining the type of the parameters of context and query messages,
classes in the properties of the messages in M_Q, and
classes representing the services to be discovered

IC_Q : classes that are directly and indirectly references by classes in DC_Q (transitive closure)

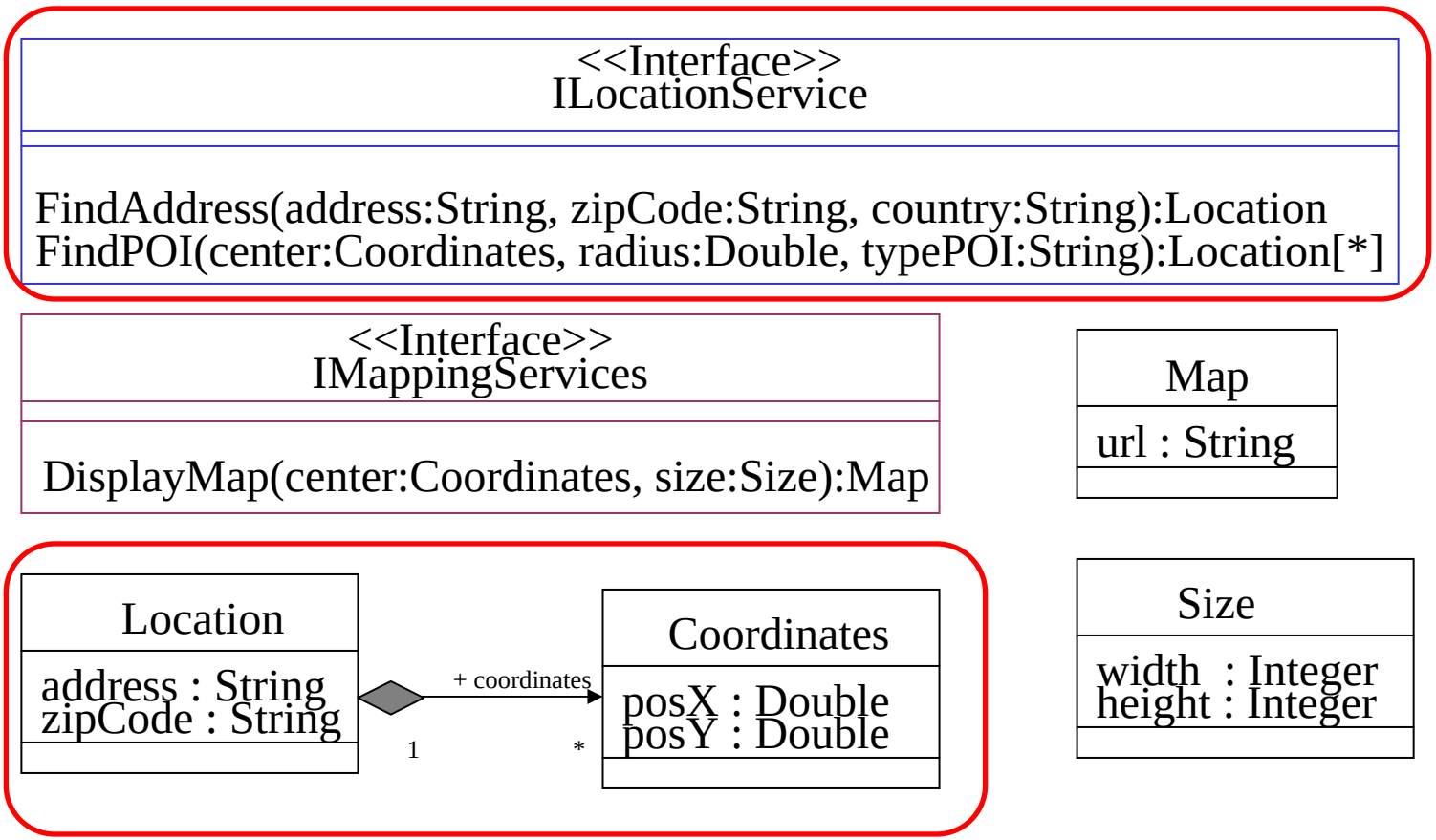
Example

Global Positioning Service Centric System



Example (SySM)

Global Positioning Service Centric System



Query Execution Engine

- Two-stage process
  - *Filtering*: search for services with operations that satisfy the hard constraints of a query
  - *Best operation matching*: search through filtered services to identify operations that have the best match with the soft constraints of a query
- Best operation matching: *graph matching algorithm* based on the assignment problem

Best Operation Matching

- Graph G:
  - V_Q = Oper(Q) U DV_k V_S = $Oper_S$(Q)
  - E(V_Q, V_S)
  - Weighted edges: D(F,v^Q_i,v^S_j), overall distance between v^Q_i and v^S_j (values [0,1])

$$D(F,v^Q_i,v^S_j) = \Sigma_{f\in F}\, w_f\, d_f(v^Q_i,v^S_j)$$
if $v^Q_i \in Oper(Q)$, $v^S_j \in Oper_S(Q)$

$D(F,v^Q_i,v^S_j) = 1$ if $v_i \in DV_k$

$D(F,v^Q_i,v^S_j) = \infty$ if v_i should not be mapped onto v^S_j

Partial Distance Function

$$d_{f=signature}(v^Q,v^S) = w_N * d_L(name(v^Q), name(v^S)) + w_{IN} * d_{PS}(in(v^Q),in(v^S)) + w_{OUT} * d_{PS}(out(v^Q),out(v^S))$$

d_L : Linguistic distance based on WordNet lexicon

d_{PS} : Distance between sets of input or output parameters

(best possible morphism between elements in the sets)

$$d_{PS}(P1,P2) = \min_{pm} \left(\Sigma_{(x,y) \in pm} \; d_P(x,y)\right)$$

$d_P(x,y)$: distance between two specific parameters

(best matching between the structures of the parameter types)

Best Operation Matching

- Graph G is constructed and D computed…
- Matching of V_Q and V_S
  - Select subset $O(V_Q,V_S)$ of $E(V_Q,V_S)$ - total morphism between V_Q and V_S and minimises function $\Sigma_{(v^Q_i,v^S_j)\in O(V^Q,V^S)} D(F,v^Q_i,v^S_j)$

    (*assignment problem algorithm*)

  - Restrict $O(V_Q,V_S)$ - edges whose distance $D(F,v^Q_i,v^S_j)$ does not exceed a threshold value D_t.

Query Results

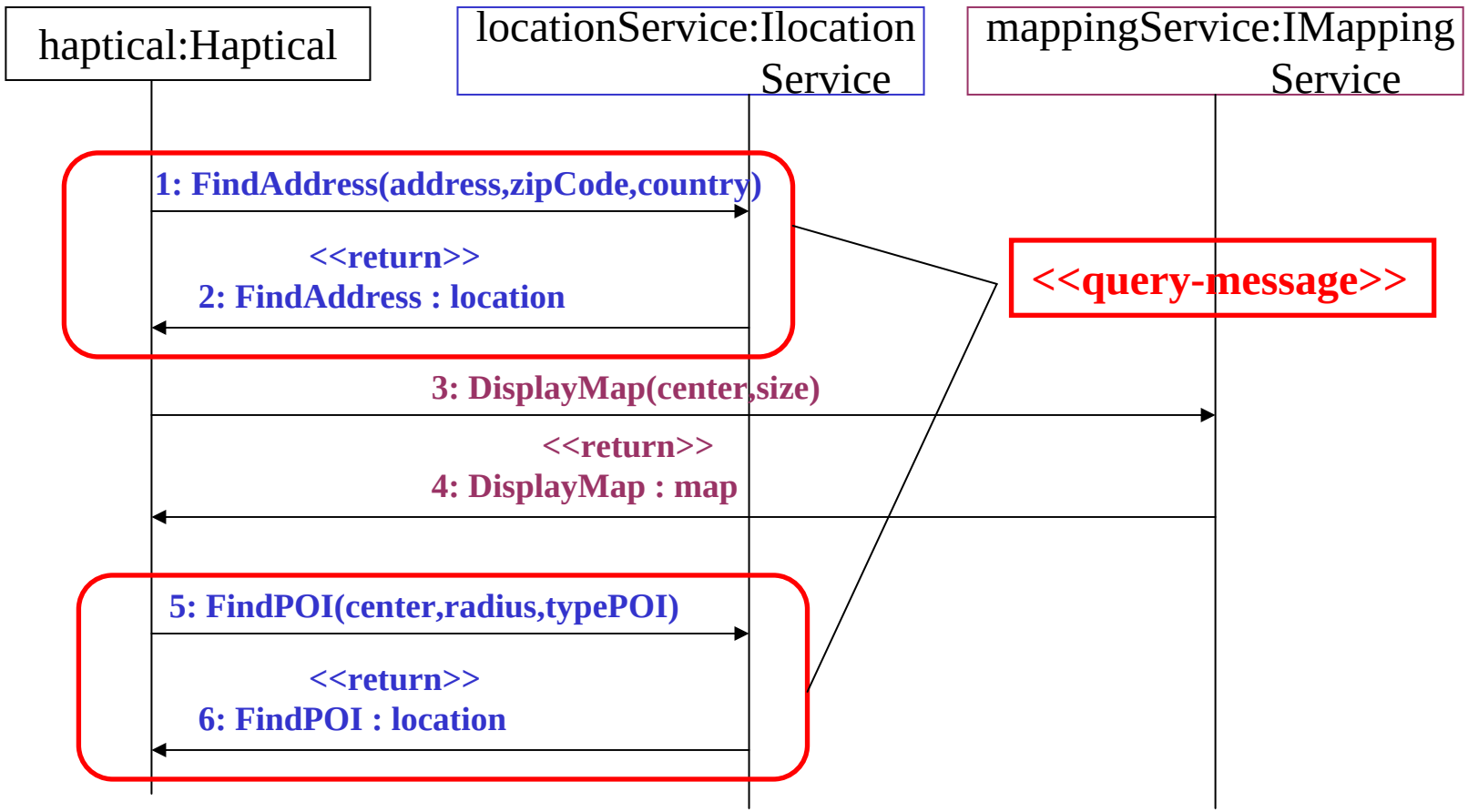
- Results are specified by *ASD profile*

ASD Result Package = Ref_I' U SySM' U service_packages

- Stereotypes of operations in service_packages
  - <<bound_operation>>: operation with the best match to a query message or selected by the designer as best candidate
  - <<candidate_operation>>: other possible results
  - <<service_operation>>: remaining operations in the service WSDL
- Query messages are replaced by bound messages
- Designers can analyse the results and select candidate operations

Example

Global Positioning Service Centric System



Example

Service registry with 45 services

FindAddress

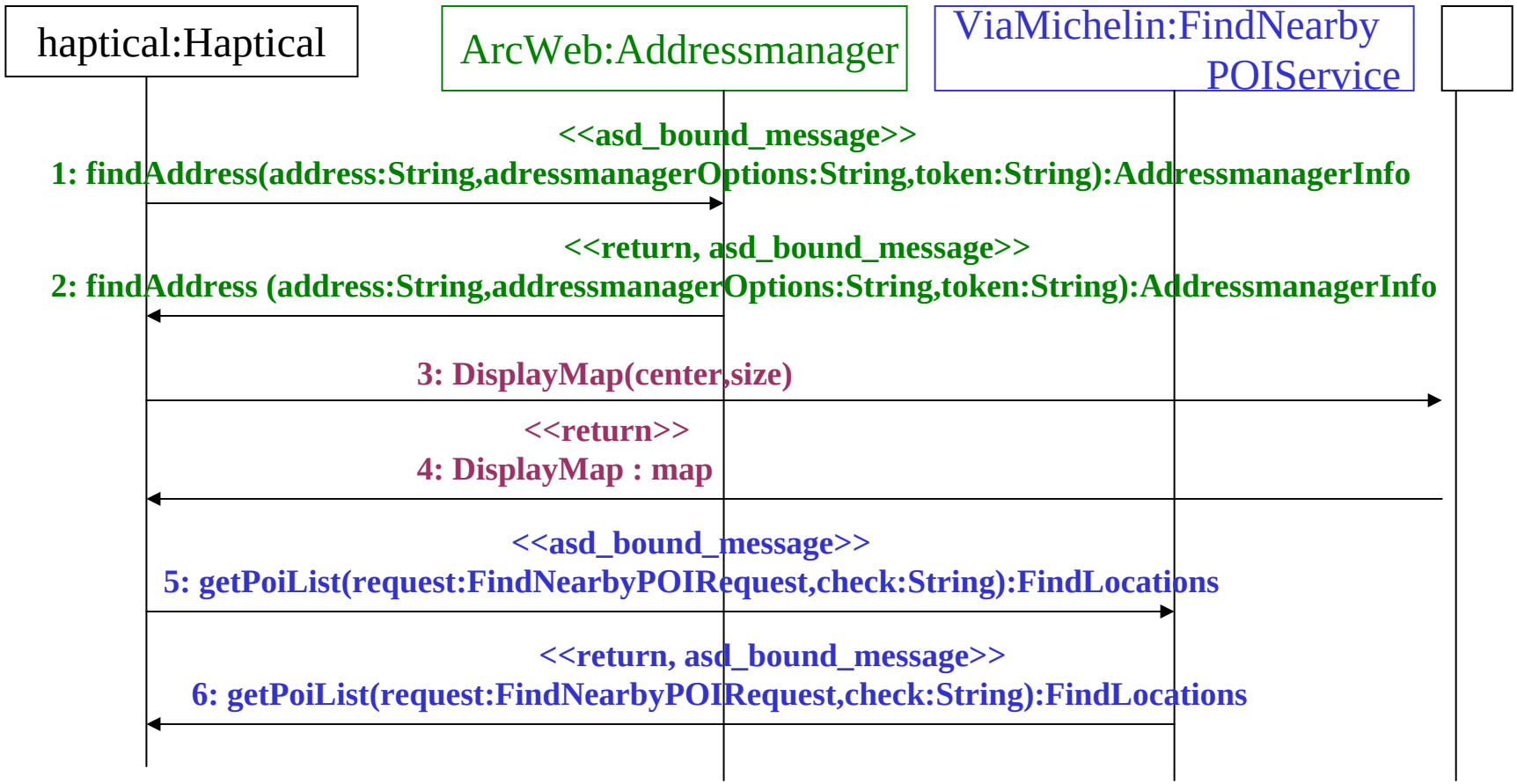
Provider	Service	Operation	Distance
ArcWeb	AdressManager	findAddresses()	0.138
cdyne.com	AddressLookup	AdvancedCheckAddress()	0.145
cdyne.com	AddressLookup	CheckAddress()	0.150
cdyne.com	AddressLookup	CheckAddressW2lines()	0.152

FindPOI

Provider	Service	Operation	Distance
ViaMichelin	FindNearbyPOIService	getPoiList()	0.136
ViaMichelin	FindNearbyPOISevice	getPoi()	0.140
ViaMichelin	FindNearbyPOIService	getCompactPoiList()	0.140
ArcWeb	PlaceFinderSample	findPlace()	0.154

Example

Global Positioning Service Centric System



Conclusions and Future Work

- Framework for architecture-driven service discovery integrated with UML-based system engineering design
- Service discovery is part of the design process of SCS
- Prototype tool as an Eclipse plug-in
- Experiments has demonstrated average precision of 62%
  (3 scenarios, 72 queries, 97 services, 1028 operations)

Currently…

- Extending and evaluating the prototype tool to support service behavioural specifications (IJWSR - to appear)
- Conducting large-scale experimentation with industrial partners
- Implementing the tool as web-services