

Semi-Formalised Data Interaction

Gary Edward Byatt 2010

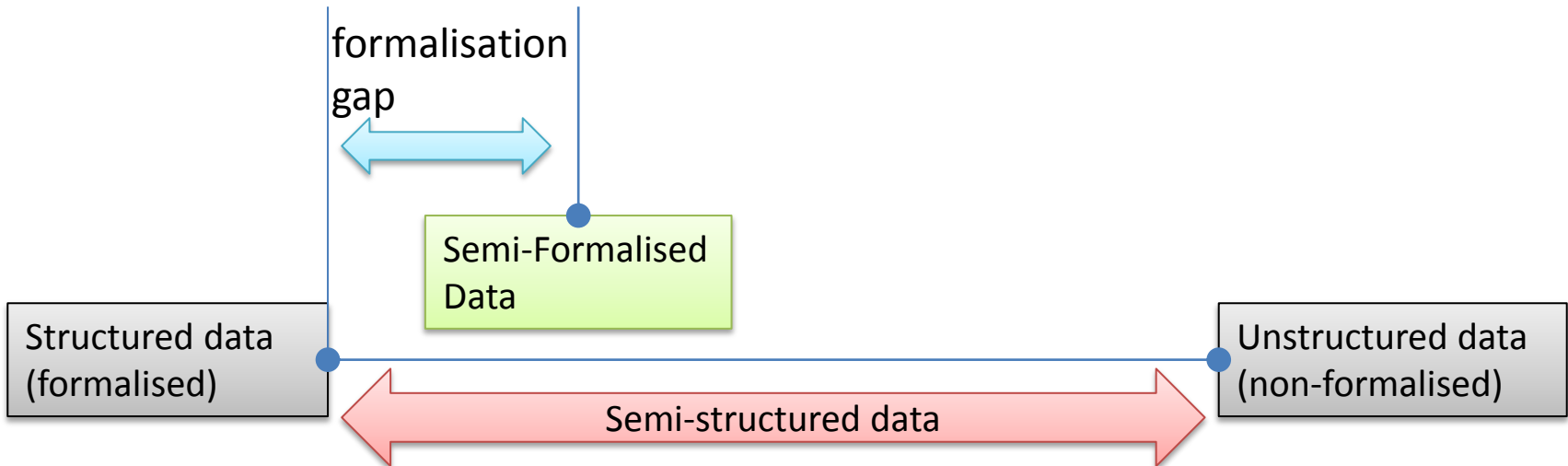
We want...

- Processes to communicate without having to code for specific pairings.
- To continue with a process pairing after changes to the form of the data communicated.

These are impossible with data exchange using published interfaces.

They are possible with Semi-Formalised Data Interaction (SFDI) in certain circumstances.

Semi-Formalism

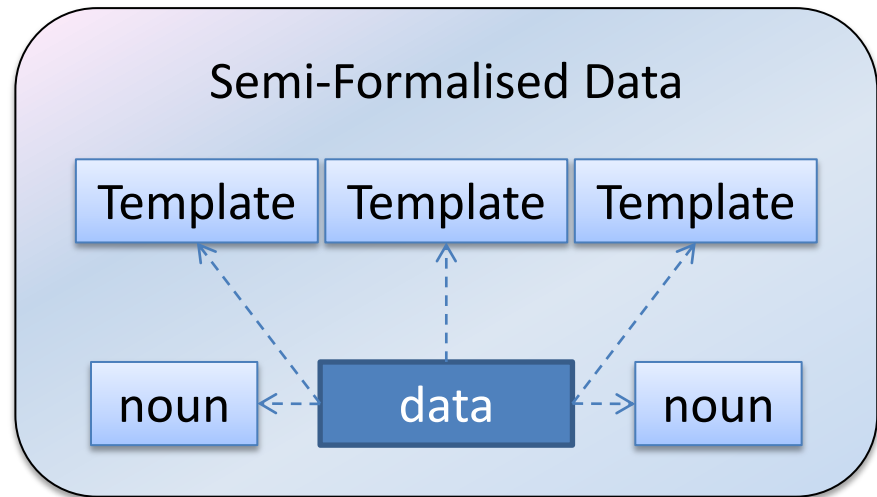
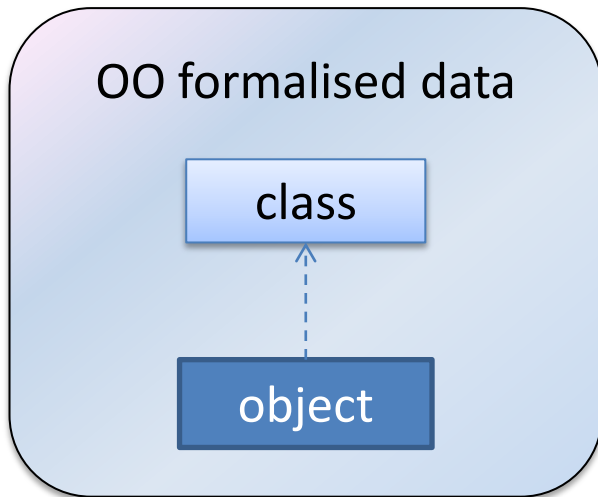


Semi-Formalism:

- Avoids problematic formalisation.
- Uses universal standards where possible.
- Data search bridges the formalisation gap at runtime.

Semi-Formalised Data

- Semi-Formalised Data (SFD) replaces dependence on one formalism (the class in the diagram) with dependence on a combination of universal standard Templates and nouns.
- Templates describe structure. Nouns are used for data labels.

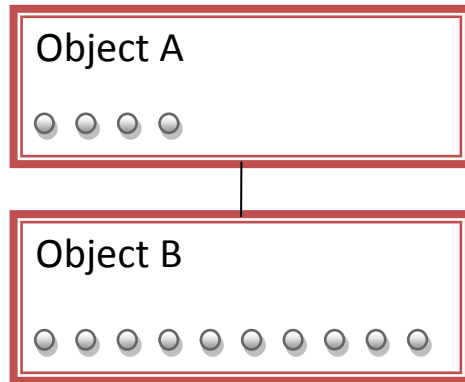


Example types of formalism

- Record types in programming languages like Pascal.
- Classes and interfaces in object oriented programming languages.
- Database schemas.
- Published interfaces such as web service descriptions.
- Some XML schemas.

Formalism and Semi-Formalism

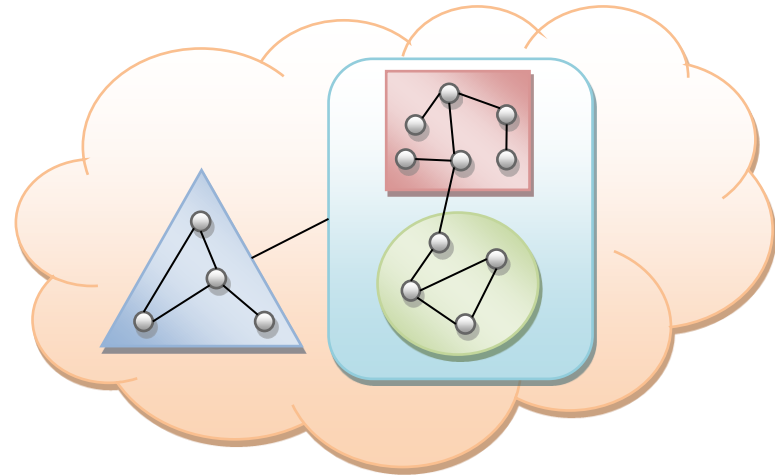
Formalised data



Formalism specifies everything:

- Attribute types.
- Attribute labels.
- Structure.

Semi-Formalised Data



Semi-Formalism uses some formalisation:

- All attributes are text.
- Universal standard nouns are suggested for attribute labels.
- Structure conforms to combinations of universal standard Templates.

Feature comparison

Formalised data

- No universal standard formalisms exist.
- Huge number of formalisms.
- New formalisms created very frequently.
- New formalisms need not be constructed from prior formalisms.
- Formalisms are mutable.

Semi-Formalised Data

- Universal standard Templates and nouns exist.
- Few standard Templates and a moderate number of standard well known nouns.
- New standard Templates and nouns created occasionally.
- New Templates must be constructed from prior Templates.
- Templates are immutable.

Assertion: Use of Semi-Formalised Data in programs is simpler and more adaptable than use of formalised data. It is also more formalised and hence more predictable than some semi-structured data.

Domain specific Semi-Formalised Data

- Like XML, Semi-Formalised Data can be specialised for specific domains, but has a richer set of semantic primitives than XML, which is limited to the tuple.
- Semi-Formalism considers universal standards to be the ideal kind of meta-data, so domain specialisations should ideally be universally standardised.
- Domain specialisation can require or suggest use of specific Templates and nouns, as well as define domain specific Templates. Suggestion is preferred to retain adaptability.
- Domain specialisation is a special case of Interaction Guide.

Semi-Formalised Data Use

- Semi-Formalised Data introduces some inherent uncertainty that is accommodated by searching for data at runtime. This is called search binding.
- Search binding has the concomitant benefit of increasing adaptability.
- Search can be specified in a deliberately 'inclusive' way to increase adaptability.

Published interface data exchange

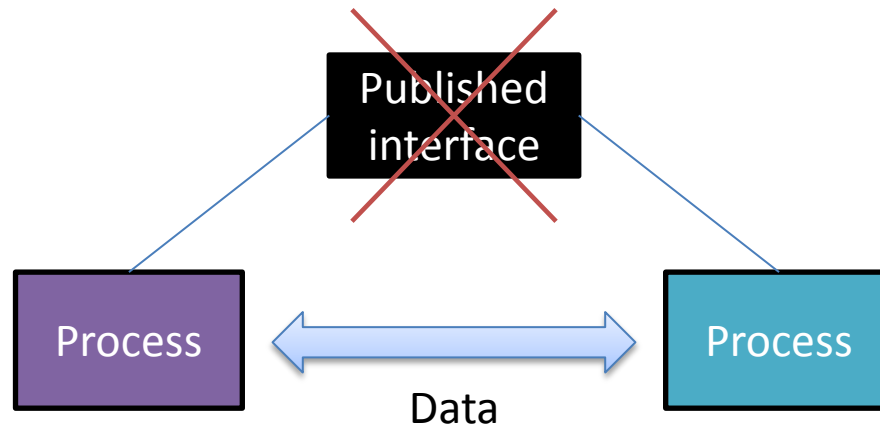
Use of published interfaces in program statements has three important disadvantages:

- Developer effort to understand the published interfaces.
- Fragile program statements due to their dependency on published interfaces.
- Mapping problems between source and target schemas.

As data is created and maintained by program statements, it too is dependent on published interfaces.

Self-describing data is directly dependent on the published interfaces which accompany the data.

Semi-Formalised Data Interaction



No dependence on published interfaces.



Advantages > difficulties, in some situations.

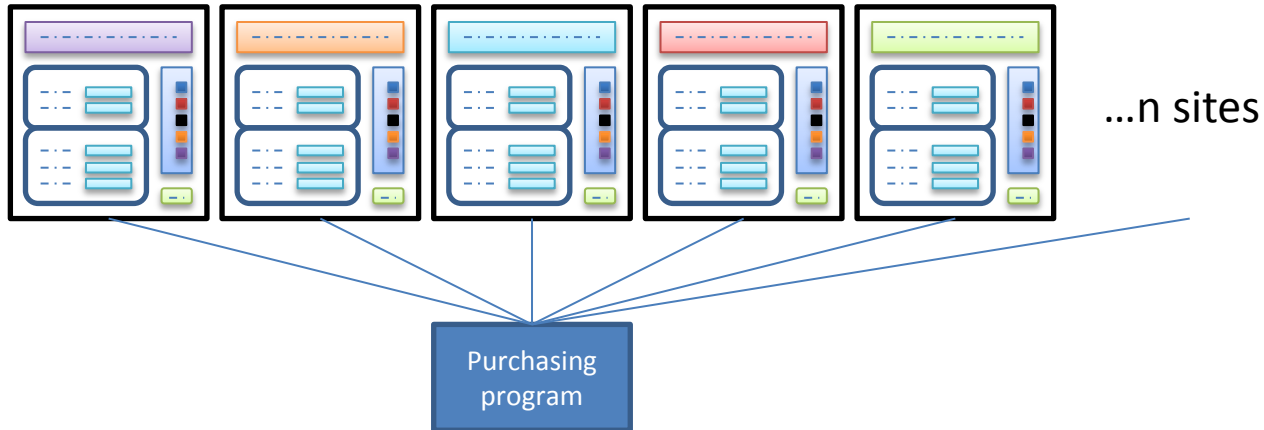
SFDI advantage situations

1. All processes that may be paired, and therefore the published interfaces that must be accommodated, are unknown at the time of programming.
2. Published interfaces that change after programming is complete.

Example use case

- The pervasive order management cycle resists formalising, yet has many recognisable parts. It is intrinsically semi-formalised and so is a good subject for Semi-Formalised Data Interaction.

Purchase from web site

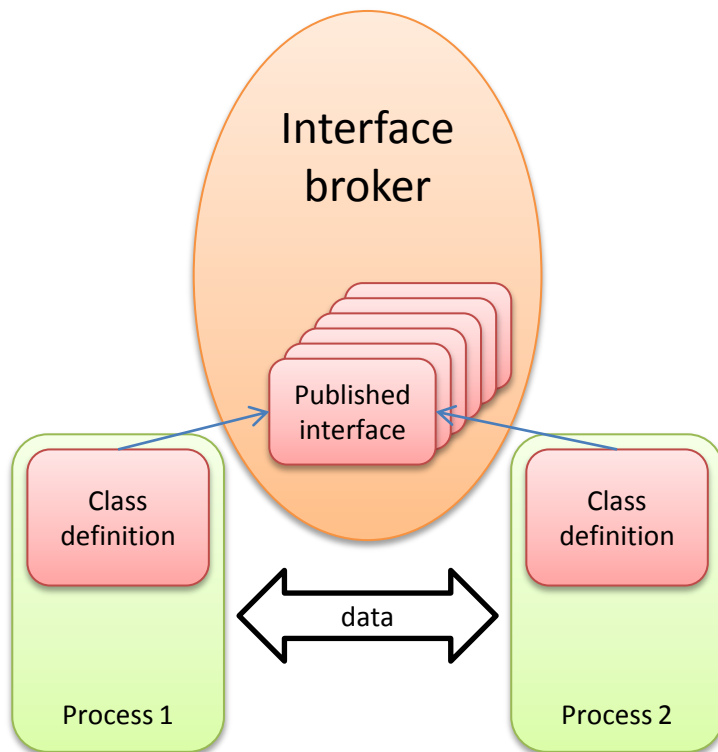


- Q No purchasing programs exist connecting to 'n' sites, but form fillers exist, why?
A Forms are semi-formalised, but the programs behind them are not.

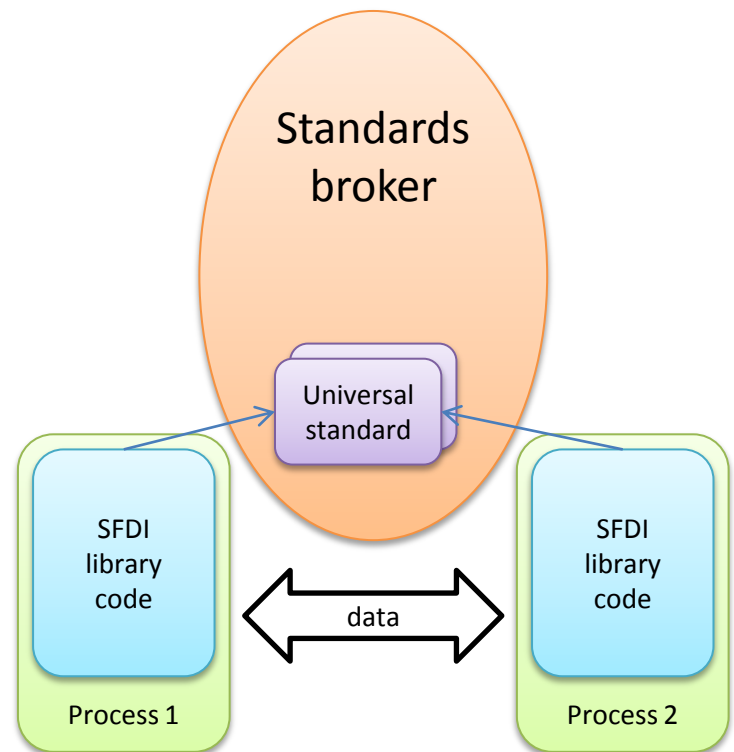
Using Semi-Formalised Data Interaction the programs could be semi-formalised.

Published interfaces and SFDI

Published interface based data exchange
Code dependence on published interfaces



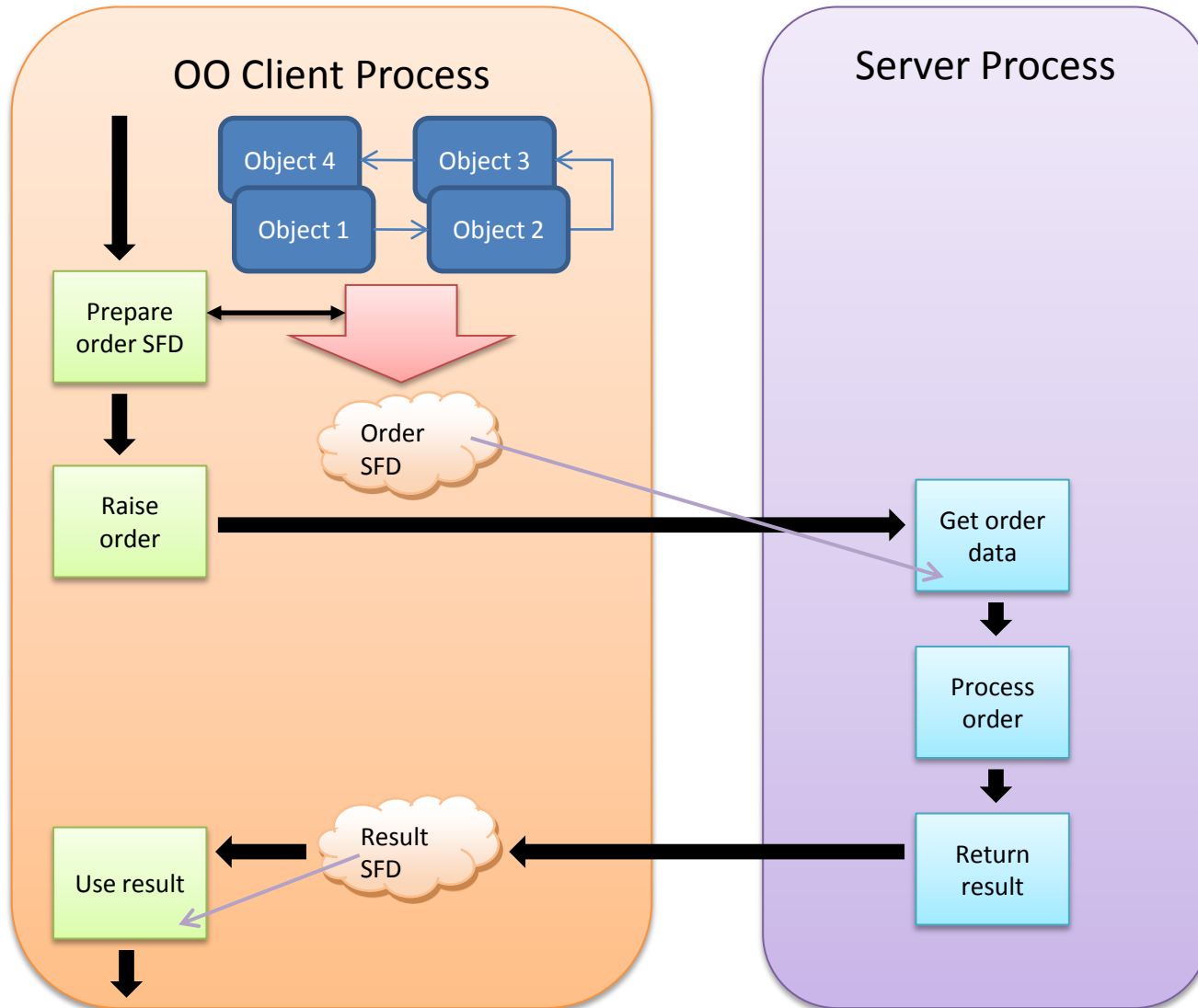
Semi-Formalised Data Interaction
Library code dependence on universal standards



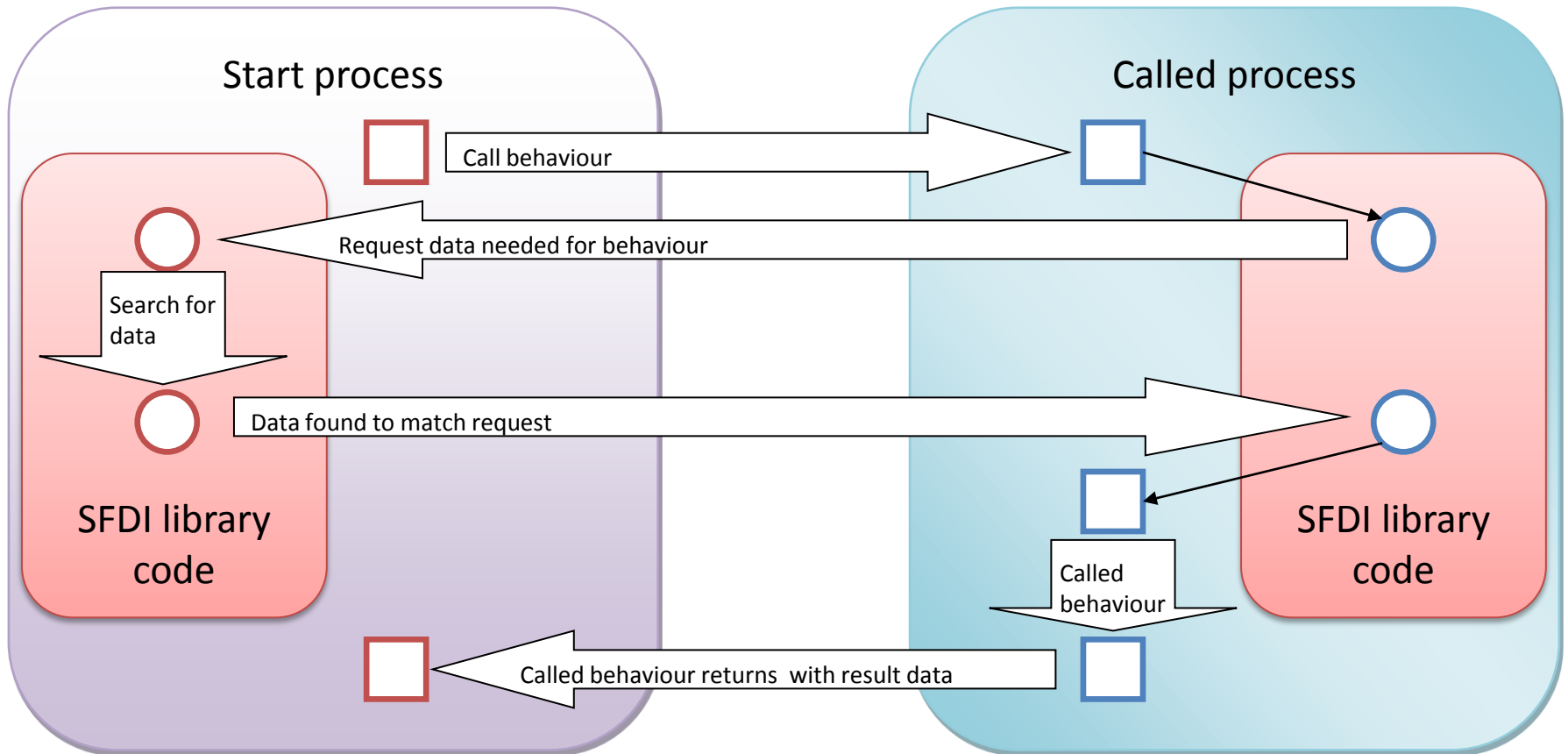
Enquiry Method


- The data consumer searches for the data it requires from the data source, i.e. data is not pushed, it is pulled from the data source by the data consumer querying the data source SFDI library code.
- This isolates the data concerns of the data consumer, obviating some data source program statements and improving adaptability.
- The data source can make use of extra in-process information to make a more informed search for the correct data than is possible with search of data pushed to the data consumer.


Raise order on service





Interaction mechanism



 Start process code

 Start process Data Manager

 Called process code

 Called process Data Manager

Difficulties for SFDI

- Accuracy of search binding i.e. correct matching of value-data associated meta-data with search text specified in program statements. Main research concern.
- Performance of search binding relative to other forms of binding. Search algorithms and hardware address this concern.

Search binding accuracy

Two classes of technique improve search binding accuracy:

- Maximise meta-data to search on, especially use of universal standards.
- Improve search – make more sophisticated use of meta-data, especially dynamic adjustment of weighting applied to search influences. Also, the data source has access to additional in-process information not available to the data consumer.

Remedial action on inaccuracy

- Alternative pairings – automated attempt at many pairings for a workable one. Should lead to multiple service versions and multiple service providers. Suitable for less critical pairing.
- Configured pairing – once per pairing at test by user and / or administrator. Suitable when a specific pairing is preferred or needed, or no alternative pairings work.
- User assist – SFDI suggests data matches, user completes / authorises them. Suitable for pairing that requires user control anyway or user assist is acceptable.
- Program modification – if a pairing is important enough and other options are not acceptable. Should involve fewer changes than coding for specific published interfaces.

Match attempts

Remedial action is simplified for cases of 'configured pairing' and 'program modification', by recording the relative perceived success of matches.

- All non-matches are equally bad and require attention.
- For multiple matches, the smallest differential between match rankings is considered least certain.
- Single matches are best, with the higher match rankings considered as better.