Information Modelling

Dr J Paul Gibson

Dept. INF

Office D311

paul.gibson@telecom-sudparis.eu

http://jpaulgibson.synology.me/~jpaulgibson/TSP/Teaching/CSC4104/CSC4104-InformationModellingPBL.pdf

Information Modelling

CSC 4104

We will look at a sequence of problems (on the board) and each will introduce important concepts when modelling information.

The problems get more complex as we progress.

We will incrementally construct our own modelling language (informally) as we require new operations (syntax and semantics) for representing structured information.

The information model acts as a bridge between the information in the ‘real world’ (requirements), and its implementation in the system (code)

We need to validate the model against the real world

We need to verify the implementation against the model

Information Modelling

Problem 1 : co-ordinates - locating things in space

Many information systems require us to locate things using co-ordinates.

We can start with a model in 2-dimensional space, but could generalise to more dimensions.

The first question to ask is the granularity of the information. Does the client/ customer use integer units or real units? What do the units correspond to in the real world?

For simplicity, let us imagine that the units are integer values describing horizontal and vertical distance (in metres) from some origin

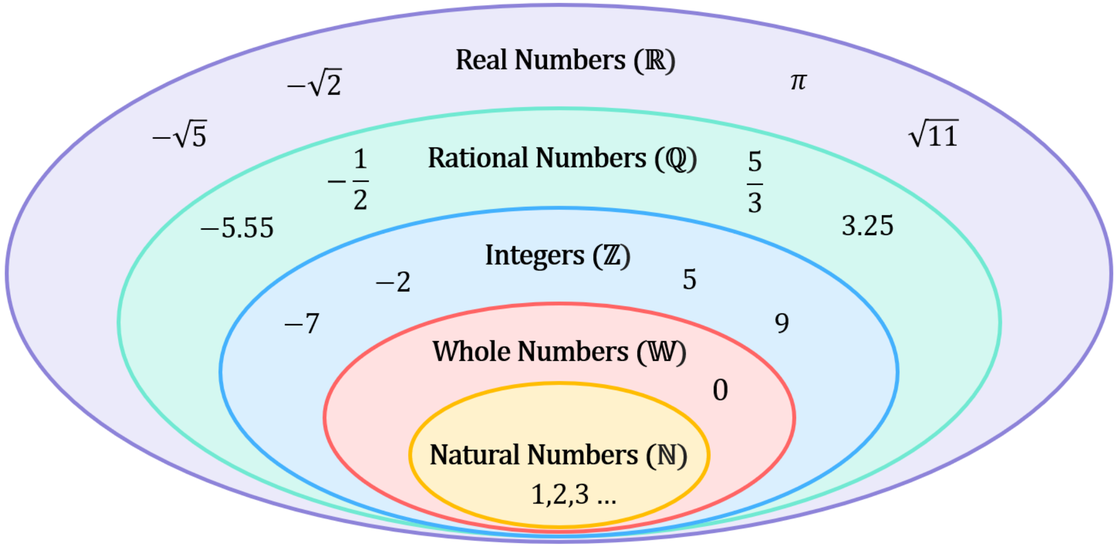
What do we need in our modelling language in order to model location co-ordinates?

Information Modelling

Problem 1 : co-ordinates - locating things in space

It would be useful to have integers (without having to construct them like we did in session 1).

We should distinguish between integers that can be positive and negative, and naturals which cannot be negative (whole numbers), and naturals which must be positive.



We also need to decide if we have bounded or unbounded values.

And, if bounded are they bounded on one side (min or max) or both sides (min and max).

And, are the bounds inclusive/ exclusive.

Information Modelling

Problem 1 : co-ordinates - locating things in space

After speaking with the client/users/domain experts, we agree that the co-ordinate values are best modelled using unbounded integers.

(Note that these may not be available in our implementation language; in which case we would have to implement them ourselves)

After we decide on the meaning of the values and their representation (let us choose decimal as that is what is what the client uses), we must decide on the structure we will use to combine (2 of) them into a single 2-D co-ordinate.

What options do we have?

Information Modelling

Problem 1 : co-ordinates - locating things in space

Some *Possible* Co-ordinate models:

An array of 2 integer elements: **int[2], [2, 3]**

A cartesian product of 2 integer elements: **int x int, (2,3)**

A set of 2 integer elements: **{x: x is an int} , {2,3}**

Question: which of these is ‘best’?

Information Modelling

Problem 1 : co-ordinates - locating things in space

A good property of a model with respect to the information it is modelling is **completeness** - can all valid pieces of information (in the real world context which we are modelling) be represented by our model?

Q**uestion: which of these is 'complete'?**

- An array of 2 integer elements: **int[2], [2, 3]**
- A cartesian product of 2 integer elements: **int x int, (2,3)**
- A set of 2 integer elements: **{x: x is an int} , {2,3}**

Information Modelling

Problem 1 : co-ordinates - locating things in space

A good property of a model with respect to the information it is modelling is **consistency** - do all values in our model correspond to valid pieces of information in the world which we are modelling

Question: which of these is 'consistent'?

- An array of 2 integer elements: **int[2], [2, 3]**
- A cartesian product of 2 integer elements: **int x int, (2,3)**
- A set of 2 integer elements: **{x: x is an int} , {2,3}**

Information Modelling

Problem 1 : co-ordinates - locating things in space

A good property of a model with respect to the information it is modelling is **non-ambiguity** - each value in our model corresponds (through interpretation) to a single unique piece of information in the world which we are modelling (ie our interpretation is unique)

Q**uestion: which of these is 'non-ambiguous'?**

- An array of 2 integer elements: **int[2], [2, 3]**
- A cartesian product of 2 integer elements: **int x int, (2,3)**
- A set of 2 integer elements: **{x: x is an int} , {2,3}**

Information Modelling

Problem 1 : co-ordinates - locating things in space

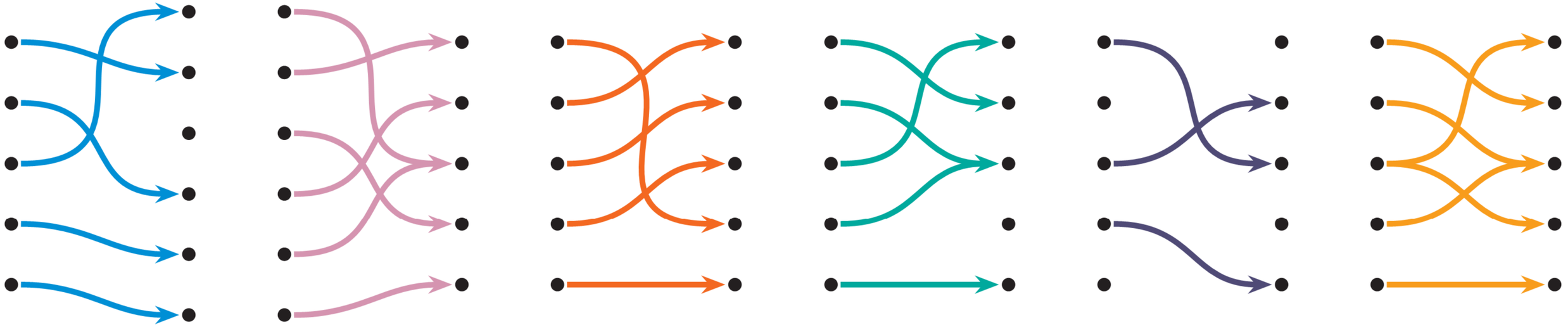
A good property of a model with respect to the information it is modelling is **non-redundancy** - each valid piece of information in our world has a single unique representation in our model.

Q**uestion: which of these is 'non-redundant'?**

- An array of 2 integer elements: **int[2], [2, 3]**
- A cartesian product of 2 integer elements: **int x int, (2,3)**
- A set of 2 integer elements: **{x: x is an int} , {2,3}**

Information Modelling

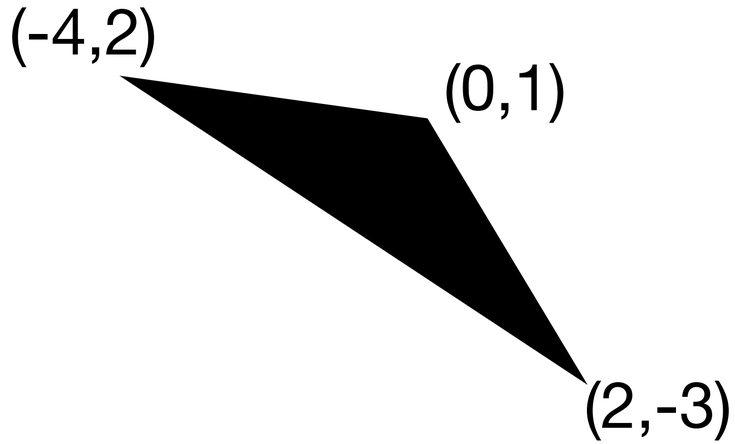
We can give these properties more mathematical (precise definitions) if we wish:



https://www.infinitelymore.xyz/p/functions

Problem 2 - model a triangle as 3 co-ordinates

Can you propose 3 alternative models and analyse them for each of the previously mentioned properties?



Note: models abstract away from information in the real-world entity they are modelling

Question: what have we abstracted away from in this example?

Problem 3 - telephone contact list

Our client wants an *intelligent* contact list in their telephone that will:

1) block incoming calls from undesirable callers,

2) take messages from unknown numbers (who are not blocked), and

3) ring for numbers that are known but not blocked.

Let us assume that we already have defined a TELEPHONE_NUMBER type that we can reuse in our contact list model.

Problem 3 - telephone contact list

What do you think of this proposal for modelling the contact information:

CONTACTS ={TELEPHONE_NUMBER} x {TELEPHONE_NUMBER}

The contact information system stores numbers in a pair of sets of **TELEPHONE_NUMBER**s

The first element of the triple is the set of blocked contacts

The second element is the set of unknown contacts

Problem 3 - telephone contact list

What about the following model?

We can use a set to represent the types of contact

CONTACT_TYPE ={BLOCKED, UNKNOWN, OK}

(**Note** that a small finite set like this one is very useful in domain modelling and most programming languages implement them using enumerations.)

Now we can model the contact information as :

{(**TELEPHONE_NUMBER** x **CONTACT_TYPE**)}

Information Modelling

Problem 3 telephone contact list

Can you propose a 3rd modelling option and compare it with the first two?

A new property to consider is whether a model is easy to update if the information it models can change. Up to now, we have implicitly assumed that the information is static, but we have to consider that information systems also contain dynamic information.

We need to be a bit more formal about what it means for a model to be “easy to update”. A reasonable definition is that a small change to the information in the system we are modelling should require only a small change to the model that represents it. (In programming terms, changes to the code should be as local as possible)

Problem 4 - Preferential elections (in Ireland)

Preferential elections allow a voter to rank a list of candidates by preference.

So a typical ballot paper once completed would look like:

Bill

Jane 2

Jill 1

Pat 3

Stephen

Here, the voter has a choice of 5 alternatives (candidates) and they have ranked their top 3 candidates.

They can rank as many as they wish: 0 is a blank vote, and 5 is a complete ranking.

Information Modelling

Problem 4 - Preferential elections (in Ireland)

Compare the following 2 information models for representing votes in an election with **N** options:

1) **Candidates [N]**, where the ith element is the candidate with preference i

2) **int [N]**, where the ith element is the preference for its Candidate

[Jill, Jane, Pat] → Bill, Jane 2, Jill 1, Pat 3

[0, 2, 1, 3, 0] → Stephen

Note, in model 2 we implicitly assume that there is an ordering for candidates (eg alphabetical)

Information Modelling

We should always aim for completeness and consistency.There are 2 main approaches - constructive and predicative

Constructive vs Predicative Data • Hillel Wayne

https://www.hillelwayne.com/post/constructive/

The two methods of satisfying the requirement for **consistent** information models are the **predicative** and **constructive** approaches.

In the **predicative** approach we have a universe of all representable items and use predicates to select a subset of that universe which we consider ‘valid’.

In the **constructive** approach we take a collection of generating rules and use them to build a set of items. As we don’t have to create a universe of data before we start building from the rules, we can make only valid data representable.

Information Modelling

It is often (always) good to model an **invariant** property for information to be valid

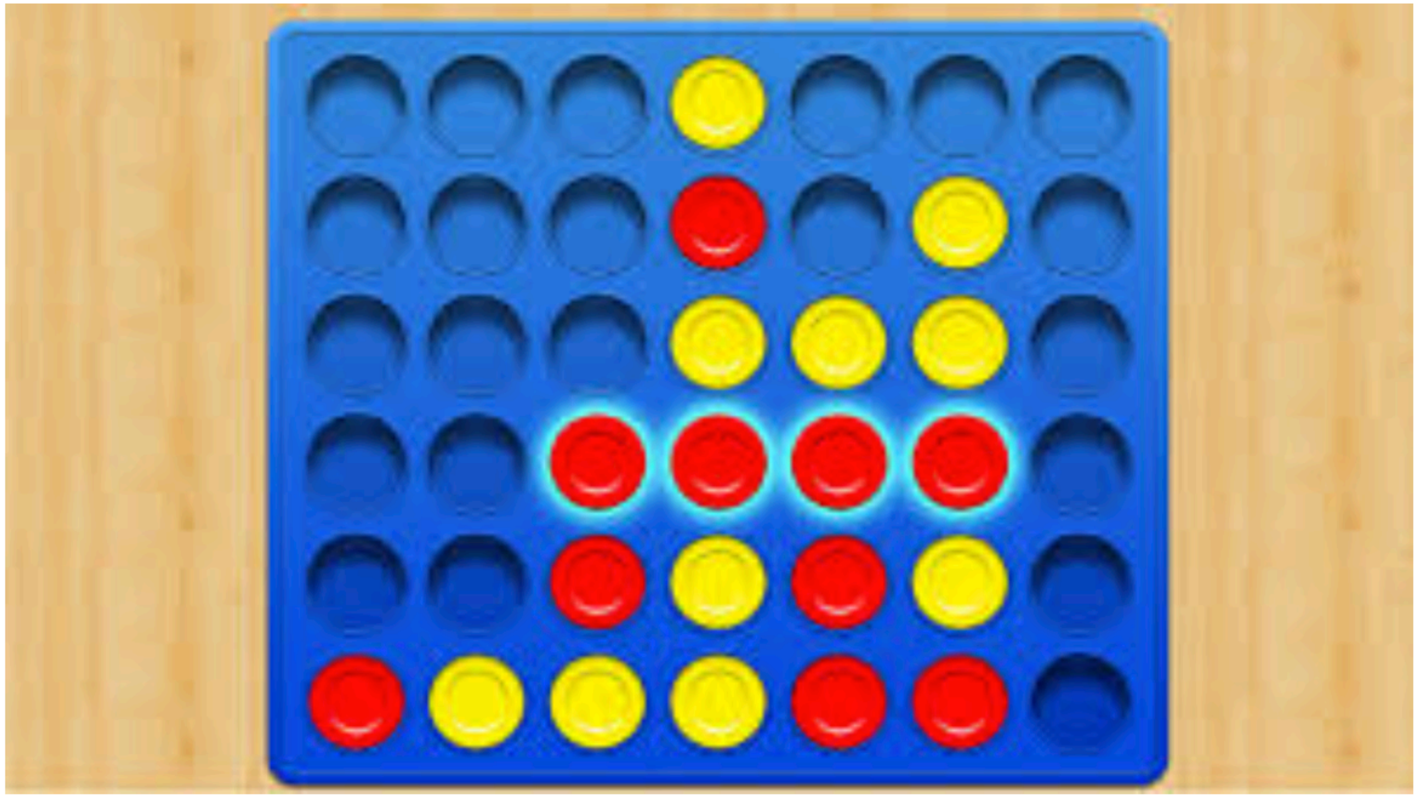
For example, in our telephone contact information model we require that contact cannot be blocked and unknown at the same time.

We can guarantee that such invariants are true by construction

We can also guarantee them by checking that they are true at run time

Problem 5 - The connect-4 game

In the game of connect-4, what are the invariant conditions that the board is in a safe state?



Write a model that represents the state of a connect-4 board.

Does your model allow the construction of invalid boards?

Can you write a complete model that guarantees validity by construction?

Information Modelling

Problem 6 - Start thinking about the SI for recording the presence of students at class

What information needs to be modelled?

How can we guarantee the validity of the information in our model?

Which information is dynamic/static?