What is Pump Facility PF101?
A Study in Ontology

Chris Partridge

Technical Report 04/02, LADSEB-CNR
Padova, Italy, June 2002

Questo lavoro è stato condotto nell'ambito dell'attività del gruppo di ricerca
“Modellazione concettuale e Ingegneria della Conoscenza”
del LADSEB-CNR

LADSEB-CNR
Corso Stati Uniti 4
I-35127 PADOVA (PD)
e-mail: mbox@ladseb.pd.cnr.it
fax: +39 049-829.5763
tel: +39 049-829.5702

MC1

METHODOLOGY: CASE STUDY 1

WHAT IS PUMP FACILITY PF101?

Issue: Version - 6.01 - 01-June-2002

For information on getting permission for other uses, please get in touch with *contact@BOROProgram.org*.

Notice of liability We believe that we are providing you with quality information, but we make no claims, promises or guarantees about the accuracy, completeness, or adequacy of the information contained in this document. Or, more formally:

THIS DOCUMENT IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT.

Contact For queries regarding this document please use the following email address:

contact@BOROProgram.org

MC1

Methodology: Case Study 1

What is Pump Facility PF101?

Contents

CONTENTS MC1

BORO

CONTENTS MC1

BORO

MC1

METHODOLOGY: CASE STUDY 1

WHAT IS PUMP FACILITY PF101?

1 Introduction

This paper is a case study that describes how the Business Object Reference Ontology (BORO) approach[1] works in practice. It describes in detail a selected part of the work using the approach that has been going on in the EPISTLE[2] community for several years. This will help people better understand not just the benefits of using the approach, but also what it is and how it is applied. It will also illustrate the kinds of results it gives - by providing specific examples of the kind of very general patterns this type of analysis typically produces.

1.1 The source of the case study

EPISTLE's prime deliverable is the EPISTLE Core Model (ECM) - a general model that is used as a basis for developing consistent application specific models. The case study is taken from an analysis of version 2.22 of the model (using an early version of the BORO approach). What sparked the analysis were some intuitively awkward questions that some of the EPISTLE users had been asking about the

[1] See *Appendix A—The BORO Program and its Approach* for details of the BORO Program and its approach.

[2] See *Appendix C—EPISTLE* for details of EPISTLE.

results of applying the framework described in the ECM v2.22 model. The analysis not only resolved these awkward questions, but also provided the basis for a simpler, more general, framework. Work is (and has been) going on to incorporate these improvements into future versions of the EPISTLE Core Model. Some of it has already been incorporated into the latest version at the time of writing - v3.1 - and further enhancements are being incorporated into v3.2, which should be published soon.

1.2 Differentiating the BORO approach

A major problem facing enterprises trying to build or integrate systems of any real richness, is trying to find a consistent semantic framework for doing this. Without such a framework, as the richness increases, so does the complexity and costs.

The EPISTLE framework was developed to provide such a framework. It, unusually in the computer industry, provides a clear high level framework – built on intuition and experience. However, as this case study reveals it is difficult to build such a framework well. Architects and users usually have an intuitive gut feeling for the general patterns that lie behind the specific details of their domain. Specifying these with the kind of clarity and exactness needed for a framework is very difficult. Architects often find themselves making clear general distinctions, which then lead to counter-intuitive and seemingly unnecessarily complicated results. These are often accepted as it is thought that it is possible to get 'good enough' results by working with and around them – and that there is no sufficiently easy way of finding a better framework.

The BORO approach provides a new tool for dealing with these situations. Like ECM v2.22 it has a framework, but one based upon a business ontology (this is one of the concepts explained in *Appendix A—The BORO Program and its Approach*). This is a relatively innovative paradigm for business modelling – and the case study shows how the ontologicallity of the framework:

- leads us towards a consistent and more intuitive model of the world. And this,

- encourages us to generalise specific patterns into simple general models – with much wider applicability.

The case study illustrates how difficult it is to regiment our raw intuitions into a consistent structure without an ontological framework. It provides examples of the typical counter-intuitive results and complexity that attempts at regimentation run into. It also provides examples of how attempts to draw out a simple general pattern from similarities based upon raw intuition can (as they usually do) lead to awkward compromises. It then shows how an ontological framework provides a consistent environment, which encourages the safe subsumption of many specific patterns into a few simple general ones – which turn out to have much wider applicability.

This has many obvious practical benefits. Intuitive, simple models are much easier to understand. Simple, general models can be transformed into simple, highly functional systems – that are easier to maintain. And in big systems, from a management point of view, simplicity and generality are a necessity.

1.3 The choice of ‘pump facility’ for the case study

The case study follows the re-engineering of the ECM v2.22 notion of a pump facility – and its associated notion of a physical pump. This recommended itself as a good choice for a number of reasons.

Firstly, it is a good example of a common problem when trying to harmonise understanding across different communities - a goal for both BORO and EPISTLE. It is a common everyday example of different disciplines using the same term, but actually talking about different (intimately related) things. The case study shows quite clearly that attempting the harmonisation raises ontological issues that can only be adequately resolved in an ontologically aware framework.

Secondly, it shows how the simple general patterns that lurk in familiar places can be extremely difficult to extract without an ontological framework. The case study provides a number of examples. One simple one is that it was (and is) intuitively obvious to the EPISTLE engineers that (what ECM v2.22 terms) pump facilities and physical pumps were both pumps. However, the general distinctions made in the ECM v2.22 framework (which generalised other important insights) meant that this simple insight could not be accommodated. The ontological resources of the BORO framework resolve this issue.

Thirdly, it reveals a general pattern that is commonplace in business models – one that it extremely likely to be re-used. Pump facility can be seen as an example of a type of role that persists through a change of its occupier – and so is, in some sense, independent of it. Another classic example is the way organisation roles – such as Chairman of the Board – are independent of the specific person occupying the role. Though in both cases they are dependent of having a pump/person of some type to 'play' the role. Clearly it is useful to have a general pattern for these types of situation that can consistently and intuitively be applied across the board.

1.4 Mechanics of the approach

The approach is based upon starting with an existing model/system – preferably a working system. This helps to scope the domain and means that a lot of the effort of formalising has been done. In the case of EPISTLE, the starting point was the EPISTLE Core Model v2.22 (from now on shortened to ECM v2.22).

The approach takes the existing model/system and re-engineers it into the BORO framework. The analysis starts with particular individuals – hence the title of the paper – 'What is pump facility PF101?'. The reason for this is our experience of the world is *grounded* in individuals – they are what we have our strongest intuitions about. We see and touch particular pumps not pumps in general.

The analysis of the individuals and the other types of objects is done on an ontological basis[3]. It focuses on a key characteristic, their extensions. How this works is explained as we start the analysis. Extension is key, within BORO, as it is the basis for an object's criterion of identity. ECM v2.22 lacks clear criteria of

[3] See *Appendix B—Core BORO concepts* for a description of the meanings of the ontological terms.

identity – and the case study shows how BORO having this makes for a better model/system.

1.5 The ECM v2.22 application model

The analysis starts by taking an application of the ECM v2.22 data model, and analyses the BORO ontology it implies. This typically involves a re-interpretation (a re-engineering) of what the items in the model originally meant – resulting in a more accurate representation with a better grounding in ‘reality’.

This case study focuses on a single simple application model to illustrate the points it needs to make. What we actually did (and what should happen in practice) was to consider a sample that was sufficiently wide to validate the elements of the framework being analysed.

An ontology is grounded through its individuals, and the BORO analysis builds up its ontology from these grounded individuals. One hurdle we had to overcome is that the formal ECM v2.22 framework has no individuals – it is a general framework for structuring process engineering applications. Individuals are informally used in the general analysis, but only formally found within these specific applications. So, to kick-start the BORO analysis, we devised a specific application model to act as an example of the things generalised into the framework.

1.6 The structure of the paper

A description of the case study is given in the main body of the paper. Ancillary information is relegated to Appendices. There is also a *BORO Working Papers - Bibliography* of referenced materials.

The case study is broken down into sections. The next section – *2 – The selected domain* - describes the domain selected for the case study – and the ontological problem it poses. *Section 3 – Analysing the ECM v2.22 domain* then provides a description of the ECM v2.22 application model for the domain. *Section 4 - Re-engineering the domain* a description of the re-engineering into the BORO framework. *Section 5 - Extending the ECM v2.22 domain* describes an extension of the domain of the ECM v2.22 application model to include the component of relation and Section 6 – Re-engineering the extended domain describes the re-engineering of this. The final Section 7 – The Case Study, very briefly summaries the results of the case study.

2 The selected domain

EPISTLE has a well-established tradition of using pumps in their examples – in particular, pump facility PF101. We follow this tradition and consider an offshore oil drilling rig (from now on shortened to 'oil rig') that has a pump (facility), PF101, specified as a component.

2.1 The context

Offshore oil rigs are large, expensive and typically built to order for use in a particular place. When designing a particular new oil rig, the design engineers specify the components, such as a pump[4], needed to construct it. And they give them reference numbers, known as tag numbers, to identify them.

When the oil rig is constructed a pump meeting the specified requirements is procured and installed. This pump will typically have been constructed at a factory to the manufacturer's specification for that type of pump – and allocated an identifying serial number.

Once the oil rig is put into operation, it is usually operated by an operator, and maintained by a maintainer. This operator thinks and works with the tag number allocated by the designer. Whereas the maintainer thinks, like the manufacturer, in terms of a manufacturered pump with a manufacturer's serial number.

This gives us two views of a pump:

[4] In practice the specification often stipulates a requirement for a number of pumps – one to be actually installed and the others kept as back up. Here we simplify things and assume that only one pump is specified and actually installed.

- The designer and operator see the pump as a component of the oil rig (the designer actually specifies it as a component and allocates it a tag number).
- The manufacturer and maintainer see both the original pump installed in the oil rig and its replacement as things that have been made in a factory and installed in the oil rig.

From now on we will call these the designer/operator's and the manufacturer/maintainer's views.

2.2 The domain

We start by taking as our initial domain a situation to which these two views can be applied. We consider the pumps that exist in the simple situation where a maintenance engineer replaces an oil rig's pump – or, put another way, where he or she takes the original pump out of the oil rig and replaces it with the new one.

To make it easier to talk about them, we name the pumps identified in these views. In hallowed EPISTLE tradition, we give the designer/operator's pump the tag no. PF101. We take the manufacturer/maintainer's pumps as having serial numbers XYZ1234 and XYZ5678, respectively.

2.3 The challenge

It may seem that all we have here is a situation where the same term 'pump' is used (by the different groups of people, with different interests) in two different senses. That all we need to do to harmonise understanding is to provide a description of what the designer/operator and manufacturer/maintainer mean – in effect, what their pumps are, and how they differ.

2.3.1 A seemingly innocent assumption

But this makes the seemingly innocent assumption that since the designer, operator, manufacturer and maintainer know what they mean sufficiently well to do their job, that this is sufficient for building these 'meanings' into a formal framework. What the EPISTLE engineers found is that this is not so. They had to work quite hard even to develop a reasonably consistent way to explain what the two senses of pump are within an overall framework. If we examine why this is such hard work, we find it is due to a lack of a clear answer to an ontological issue: What kinds of things are the two senses used to describe?

2.3.2 The core ontological issue

To appreciate this issue consider the following situation, that unfolds in three stages.

1 Consider a meeting where we show representatives from the operator and the maintainer around the oil rig and ask them to point to their pump (PF101 and XYZ1234, respectively). They clearly point to the same thing, a pump installed in the oil rig. Questioning does not seem to reveal any discrepancy. For example, they agree on the pump's operating characteristics – such as flow rate, head and power. It would seem as if they are just using different names – tag PF101 and serial no. XYZ1234 – for the same thing.

   It seems tempting to say that the maintainer and operator's pumps are the same thing. Currently some engineering computer systems even insist that they are effectively the same[5]: the pump referred to by its tag no. is given the characteristics of the manufacturers pump that is chosen as the first for the job. This is a natural, intuitive, interpretation of the situation.

2 An hour later, a maintenance engineer changes the pump.

3 Two hours later, the same representatives are taken to the same place on the oil rig. A simple question now reveals a problem. If we were to ask them whether this pump was the same as the one they saw a couple of hours ago, then they would give different answers. From an operator's point of view it is the same – it is the same component of the oil rig, with tag no. PF101. From the maintainer's point of view it is clearly different; it has a different serial number, XYZ5678, and it was made at a different time and place from the earlier pump (serial number XYZ1234). The logic of this argument now tempts us to say that the maintainer and operator's pumps are different things.

But here logic leads us to the counter-intuitive conclusion that two different pumps are in the same place at the same time. Where, when we touch the pump,

[5] I am indebted to Jan Sullivan of POSC-CAESAR for this point.

we are touching two pumps even though it feels like one. We need to explain what is hopefully only an apparent contradiction - why:

> When faced with the pump our intuition seems to tell us there is only one pump there in front of us, but
>
> When considering the situation before and after the replacement, logic seems to tell us that, in each situation, there are (apparently) two different pumps there, in exactly the same place.

2.3.3 Can two things be in the same place at the same time?

Pumps with tag numbers and/or serial numbers may be relatively new, but the problem faced here is ancient. It is an age-old philosophical (ontological) conundrum usually captured in the question: can two things exist in the same place at the same time?[6]

It is important to recognise that the problem is not academic, but a practical commonsense one that the engineers working on ECM v2.22 intuitively recognised as they were building their framework. What forced it to their attention was need for clarity in the specification of that framework. And for them it was a serious practical problem. If their specification was inconsistent, then it could not serve its purpose as a framework for storing and automatically exchanging information. The engineers needed a solution.

3 Analysing the ECM v2.22 domain

We illustrate the implications of ECM v2.22's solution by constructing an example application model. This highlights the EPISTLE engineer's concerns, which then provide a motivation for the BORO re-engineering. This we do, first at the individual level and then at the upper levels.

[6] It can be traced back to one of the oldest questions in philosophy raised by the Ancient Greek Heraclitus of Ephesus in the 5th Century BC who asked "can we step into the same river twice". The river's water changes much as the designer's pump changes manufacturer's pumps.

3.1 The ECM v2.22 application model

To give a context for the construction of the application model – we start by looking at the general approach taken by ECM v2.22 towards solving the ontological issue.

3.1.1 ECM v2.22's solution to the ontological issue

The EPISTLE engineers spent a lot of time thinking about this issue – though they understandably did not think of it as an ontological issue, more as a modelling problem. They tried out a number of options in their search for a solution. By the time ECM v2.22 was issued, their proposed solution was to make an absolute distinction between two types of thing. They suggested that the manufacturer/maintainer's pump is a physical thing and the designer/operator's pump is a logical thing (more specifically a facility, which is a subtype of functional thing, which in turn is a subtype of logical thing) where physical and logical things are quite distinct (these subtypes can be seen in *Appendix D*'s *Figure MC1–0.1*). The relationship between the two is that the physical manufacturer/maintainer's pump is installed in (and deinstalled from) the designer/operator's pump facility.

Effectively ECM v2.22's answer to the question - "Can two things be in the same place at the same time?" - is a limited 'yes'. It says: a physical thing and a logical thing can be in the same place at the same time – but they are there in a different way. In the situation that gave rise to the question, the manufacturer/maintainer's pump is physically and the designer/operator's pump logically there. And for things like pumps, these are the only two ways in which they can exist – and so co-exist in the same place at the same time.

EPISTLE's engineers' intuitive concerns

This solution 'worked' well enough, though it required some odd workarounds. But, some of the EPISTLE engineers were not really satisfied. They felt that a number of their intuitive concerns – particularly with regard to the status of logical objects – indicated that they were missing some important opportunities to improve the framework. All in all they felt that this solution was not sufficiently 'right' for what they needed to do. The nature of these concerns will become clearer after we construct the application model for our example – which we start doing now.

3.1.2 Constructing the application model

We follow the normal BORO approach and start our analysis with individuals (in ECM v2.22's term's instances – that is, belonging to the instance subtype - not entity instances, which are something different). This means we build the application model from the bottom up, starting with the individuals then moving onto the individual's relations with other individuals (in ECM v2.22's term's association instances) and finally – in the next section – taking in the more general classes.

The individuals

As explained above, the ECM v2.22 tactic for resolving the 'two pumps or one' issue is an absolute distinction between physical things and (logical) facilities. This means that in its ontology the different types of pumps fit under the distinct physical and logical things subtypes on the subject dimension of its framework.

The designer/ operator's pump

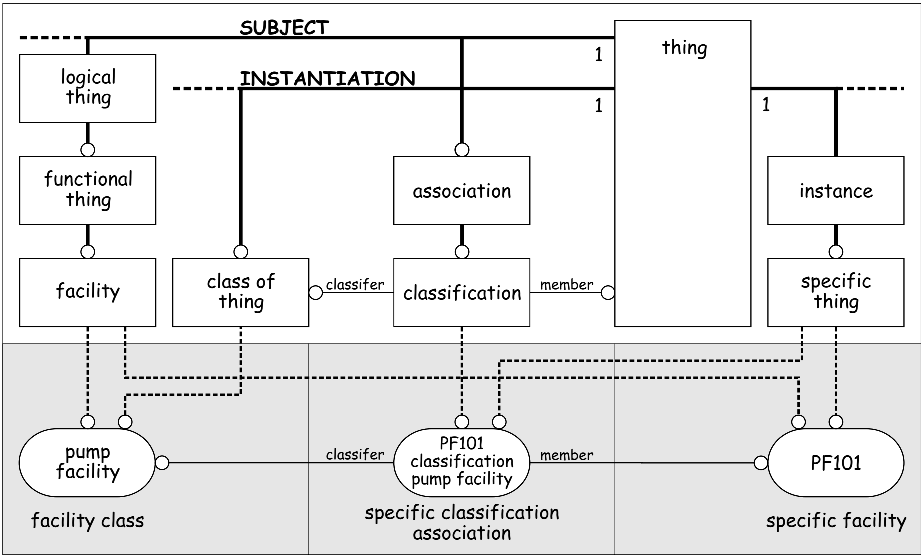
The designer/operator's pump is a pump facility, tag number PF101, fitting under the subtype, logical thing. Following ECM v2.22's rules that each entity must belong to a subtype from each of its framework's dimensions, PF101 fits under the logical (subject), instance (instantiation), actual (life-cycle), real (reality) thing entity subtypes. We are not interested in the life-cycle and reality dimensions in this example, so we ignore them from now on.

For various reasons[7], ECM v2.22 uses two types of 'instantiation' – the EXPRESS 'instance of' relation and the ECM v2.22-defined 'class-member' association. Typically both of these are used when 'fitting' instances, such as PF101, into the framework.

[7] The principle reason is that EXPRESS language does not directly support classes of classes. The class sub-type and the class-member association are a work-around that enables EPISTLE to have classes of classes. However it introduces an issue about the nature of entities such as facility. Is it a class whose instances are facilities – or a class of classes of facilities? However these questions are outside the scope of this paper. Note, however, that BORO only has a single 'instantiation' (class-member) relation, and so avoids these problems.

So PF101 is, firstly, an instance of two entity types - specific thing and facility. This tells us that PF101 is a specific facility. PF101 is, secondly, a member of the pump facility class (this necessitates 'introducing' the pump facility class for it to be a member of). This tells us it is a pump facility. Together they tell us PF101 is a specific pump facility. The model in *Figure MC1–1*[8] below gives us a picture of this.

Figure MC1–1 Pump facility PF101



Using two types of instantiation here leads to some duplication - as both types tell us PF101 is a facility. This creates the possibility of a contradiction, where something is classified under different, mutually exclusive, subtypes by the two types of instantiation. This is avoided by the ECM v2.22 principle that "[t]he *classifier* and *member* must both be of the same *subject* subtype"[9] – in this case ensuring that both the pump facility class and PF101 belong to the same subject type – facility[10].

[8] This, like all the other EPISTLE figures in this paper, uses the EXPRESS-G notation extended to handle instances of entities

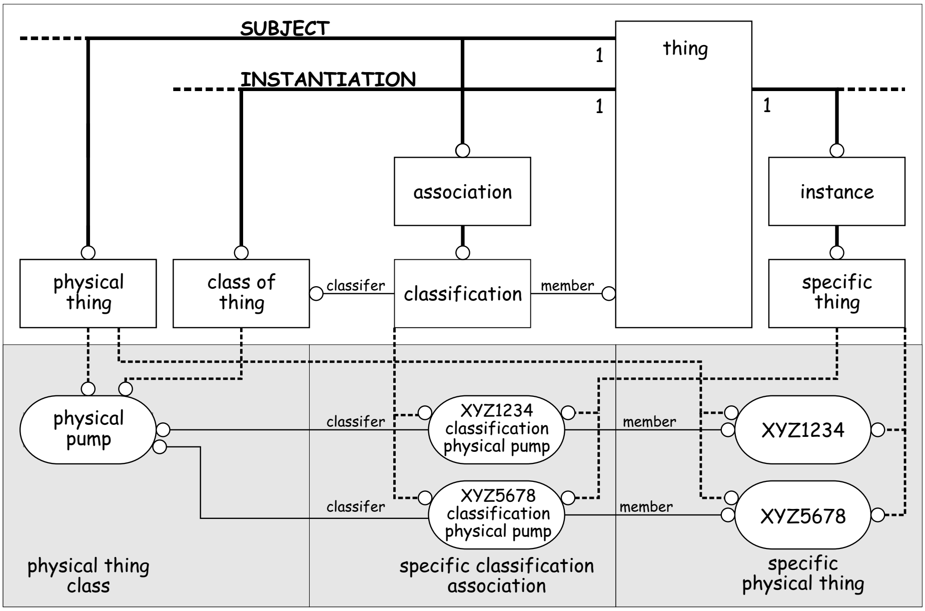
[9] §6.5.5 Classification, *EPISTLE Framework V2.22*.

[10] Architectually this duplication indicates that the distinctions in the framework do not quite align up with what they are going to describe. However, much of the unnecessary structure here is a result of the less than satisfactory EXPRESS language (which EPISTLE had to work within) rather than a result of the engineer's analysis.

The manufacturer/ maintainer's pumps

There are two manufacturer/maintainer's pumps; the original physical pump, serial number XYZ1234, and the new physical pump, serial number XYZ5678. ECM v2.22 fits these into its framework in an analogous way. They are instances of the physical and specific thing subtypes – and members of the physical pumps class. The result is modelled in *Figure MC1–2* below. They also 'fit' under the actual (life-cycle), real (reality) thing entity sub-types – in ECM v2.22's other two dimensions – but, as mentioned before, we ignore these here.

Figure MC1–2
Physical pumps XYZ1234 & XYZ5678



It is worth noting in passing that the ECM v2.22 framework does not allow a single 'pump' class to be constructed with both physical pumps and pump facilities as members. It 'forces' you to have separate physical and logical classes for each of them. (This is not as clear as it could be because we have modelled the two classes in separate diagrams.) This counter-intuitive restriction is a result of the rule mentioned above where classifier and member must belong to the same sub-type and its use of separate physical and logical subtypes[11].

[11] This is a simple example of how trying to make clear general distinctions can have obviously counter-intuitive unintended results. In the BORO analysis the pump class returns.

The installation associations

Figure MC1–1 and *Figure MC1–2* show the three individuals (in ECM v2.22 terms, specific instances – excluding associations) in our example – PF101, XYZ1234 and XYZ5678. We now need to specify how these are related (in ECM v2.22's terms, the specific instances of associations that relate them).

At this stage the only (sub)type of association in our scope is installation. And there are only two associations: physical pumps XYZ1234 and XYZ5678 both have, at different times, an *installation* association with pump facility PF101.

Activities beginning and ending associations

Within ECM v2.22, associations typically have life spans, and in recognition of this activities are used to mark an association's beginning and ending[12]- and so its lifespan. This is needed for the installation associations, which only last for a fixed period of time. They are 'begun' by an install activity and 'ended' by a deinstall activity.

The PF101 installation associations

PF101's installation activities help to make its associations clear. The original pump XYZ1234 underwent an install activity that began an install association between it and pump facility PF101, and undergoes a deinstall activity that ends this association. Then the new pump XYZ5678 undergoes an install activity that begins a new install association between it and pump facility PF101.

We start by modelling these two installation associations in *Figure MC1–3* below:

- Physical pump XYZ1234 has a specific installation association with PF101, and
- Physical pump XYZ5678 has a specific installation association with PF101.

[12] "An *association* has a *lifetime* during which the association is a valid one. Conceptually, the start and end points of the lifetime of an association are defined by the activities that bring the association about and terminate it." §4.3 Association, *EPISTLE Framework V2.22*.

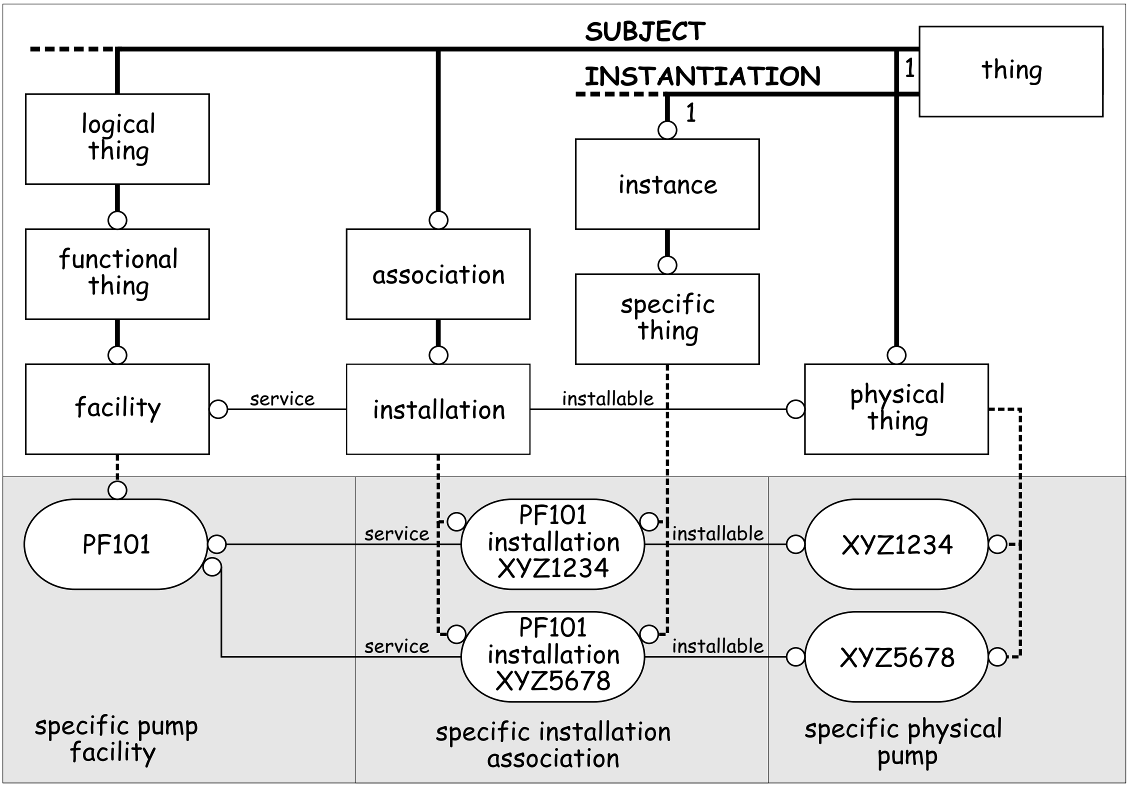


Figure MC1–3
Pump specific installation associations

The activities

We now consider the activities. There are two types of activity: installation and construction. We first consider the installation activities associated with the installation associations. Then we look at the construction activities, which have no associated associations.

PF101's installation associations' install and deinstall activities

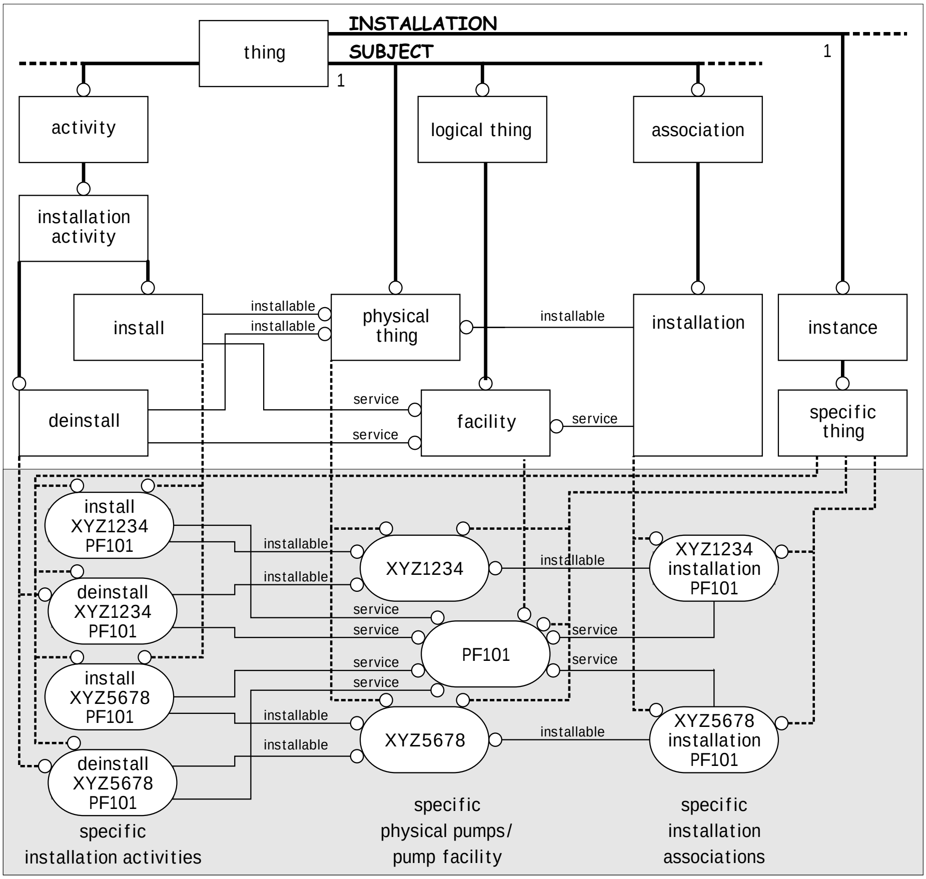
The two installation associations in *Figure MC1–3* each have their own install and deinstall activities, to mark their lifespan – giving us these four activities:

- Install XYZ1234/PF101,
- Deinstall XYZ1234/PF101,
- Install XYZ5678/PF101, and

- Deinstall XYZ5678/PF101

We keep the model simple by excluding a couple of things. Firstly, replacement activities, which are just a deinstall activity followed by an install activity[13]. Secondly, the event effects (a subject subtype) that ECM v2.22 normally uses to link the start and end activities directly to their association. This means that in our model the activities are indirectly linked to the association via the things that participate in it – as can be seen in *Figure MC1–4* below.

Figure MC1–4 Pump facility PF101 installation activities



[13] "We would expect to have a pair of *activity* entity types corresponding to each *association* entity type. In addition, many associations have an additional *activity* entity type that combines the effect of the previous two activities into a single activity (or transaction)." – §4.4 Activity, *EPISTLE Framework V2.22*

Physical pumps construction activities

For the BORO re-engineering it will help to have a model that marks out the extent of the lives of the physical pumps – their beginnings and endings. We can consider the activity that begins their lives (following a hint in *EPISTLE Framework V2.22*[14]) as a construct activity in a factory composed of a series of (ECM v2.22) assemble activities – one for each part, which 'begins' an assembly association between the pump and the respective part. Similarly the activity that ends their lives would be a dismantling activity composed of a series of disassemble activities – 'ending' the assembly associations.

For the sake of simplicity, we just model the overall construction activity, with construct and dismantle subtypes – without concerning ourselves with the series of assemble and disassemble activities of the individual parts or their assembly associations.

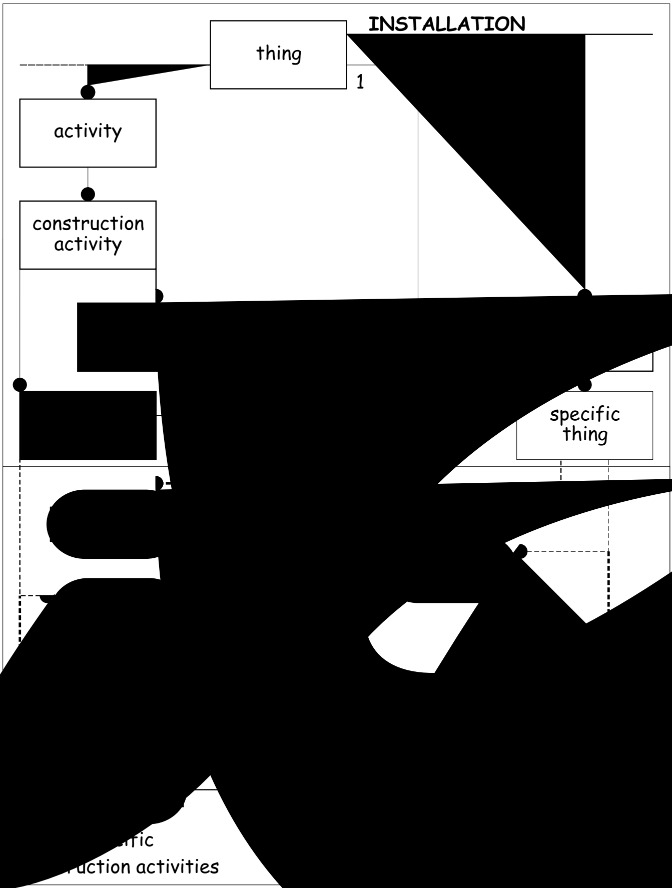
This gives us these four construction activities:

- Construct XYZ1234,
- Dismantle XYZ1234,
- Construct XYZ5678, and
- Dismantle XYZ5678.

These are shown in *Figure MC1–5* below.

[14] "In addition, … an additional activity entity type that combines the effect of the previous two activities into a single activity (or transaction)." §4.4 Activity, *EPISTLE Framework V2.22*.

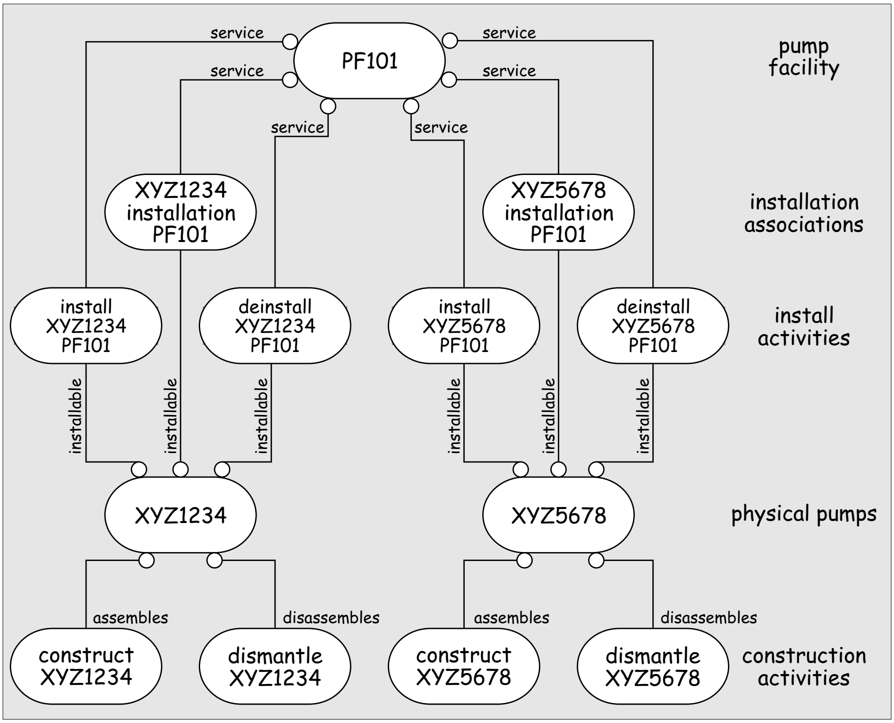
Figure MC1–5 Physical pumps construction activities



ECM v2.22's ontology of individuals

We now have a sufficiently [illegible] cation. There are eleven specific things in the gro[illegible] grammed in *Figure MC1–6* (below). These are the things th[illegible] in 'reality' and which the BORO re-engineering will take a[illegible] To help us focus this analysis, we now clarify the intuitive c[illegible] ngineers using ECM v2.22 had with this ontology.

Figure MC1–6
The example's ground level ECM v2.22 ontology



3.2 EPISTLE engineers' intuitive concerns

The engineers using ECM v2.22 had a number of commonsense concerns about this way of characterising their world – easily expressed in simple questions, such as:

> When the physical pump XYZ1234 is pumping, what is the logical facility PF101 doing?

Or even more basically:

> When I touch physical pump XYZ1234, am I also touching logical facility PF101?

These show how intuitively clear it is that the pump facility and the physical pump are inter-related in all sorts of ways - much more than just their install and de-install activities. The root of the engineers' unease seems to be the lack of a clear explanation of these inter-relations, in the context of the overall absolute physical/logical distinction.

These can be seen as a lack of a clear structural, architectural (ontological, even) framework for the relationships between physical and logical objects occupying the same place (i.e. co-extensive) and:

- the properties they have,
- the activities they participate in, and
- the parts they have.

We examine these now.

3.2.1 Co-extensive things' properties

One of the basic ways in which two co-extensive pumps are related is by their properties. The two pumps share many of the same types of properties. And when a physical pump is installed in a pump facility (and so they become co-extensive), they necessarily have many of the same properties – e.g. the same height, the same weight.

ECM v2.22 effectively offers little in the way of explaining how properties are related in general. In its model, Properties are a subtype of Characteristic, which is a different subject type from Physical and Logical Object (you can see this in*Appendix D*'s *Figure MC1–0.1*). In its terms, any 'Thing' can have a 'Possession Association' with any type of 'Characteristic'[15] – and there is no division of Characteristic or Property into physical and logical. There are no general constraints on properties and their relationship to Physical and Logical Things.

To understand the nature of the engineers' concerns it is worth examining the options for providing a better explanation within ECM v2.22's absolute physical/logical distinction.

[15] A thing may possess specific characteristics, modelled by a possession of characteristic association linking the characteristic(s) to the thing(s) possessing them. … Note that the thing being described by … possessing a characteristic may be of any subject subtype." §7.2 Activity, *EPISTLE Framework V2.22*.

Distinguishing physical and logical properties - I

At the architectural level, one of the first things that needs to be clarified is whether the absolute physical/logical distinction extends to properties.

One could stipulate that there are distinct physical and logical versions of many properties. But when we consider properties, like tangibility and visibility, which seem particularly physical, this does not seem viable. In fact, it seems just contradictory to talk about logical tangibility and logical visibility.

Anyway, even if we were to find a satisfactory way of explaining how there are distinct physical and logical versions of properties – and so creating parallel physical and logical worlds of objects and their properties – we would be no closer to explaining the relationships, including necessary sameness, between the properties of co-extensive physical and logical objects.

So the option of extending the distinction to properties is unattractive. It seems intuitively wrong and provides no answers to our question about the inter-relationship of the properties.

Distinguishing physical and logical properties - II

The physical/logical distinction may apply to properties in another way. It may be that instead of there being different physical and logical versions of properties (as there are pumps), that some properties are irredeemably physical or logical.

This seems to be the case with 'physical' properties, such as tangibility and visibility. It seems intuitively obviously that these can only be properties of physical things. Surely if something is tangible and visible, it is physical.

The issue then arises whether logical things can have 'physical' properties . Could a logical pump be 'physically' visible and tangible? If we say yes, this raises more questions. Presumably it would be seen and touched in the much the same way as the physical pump? In fact, whenever we actually see or touch one, we would do so in exactly the same way. We can only see and touch a pump facility when it has a physical pump installed, so seeing and touching a logical pump involves seeing or touching a physical one – though not obviously vice versa.

But there is the inter-relationship between the pumps' properties to explain. Why do a pump facility and its installed physical pump necessarily have the same properties? Maybe the two pumps are sharing their tangible property – that is, there is one tangible property that they both have (to confirm this it would be helpful to have a criteria of identity for properties). But, if so, how does this work? What happens to this shared property when the physical pump is deinstalled?

These considerations may make the option of restricting physical properties to physical objects attractive. But this option has its own problems – especially for things like pump facility PF101 that we 'naively' think we can see and touch (and occupies space and time). If logical things cannot have these types of physical properties, how do we know about them? What happens if we cannot, in principle, see and touch (or otherwise perceive) the logical pump PF101 (if only the physical pump has the property of being visible and tangible)?

In trying to interpret ECM v2.22, we are torn between a notion of logical object that precludes physical properties and the need to explain how we experience things such as pump facilities. And the framework, as it was constructed, does not give us an answer.

What is logical existence?

This is closely related to another intuitive discomfort felt by the engineers. They wondered what happens to pump facility PF101 in the gap between the deinstallation of physical pump XYZ1234 and the installation of physical pump XYZ5678. Does it carry on existing? And on what basis should they make the decision?

If the logical pump has no obvious physical properties, then these cannot be the basis for establishing whether it exists. One option to consider might be that the existence of PF101 depends upon the existence of an installed physical pump. But, it seems odd that the existence of a logical thing like PF101 should rely on the existence of a physical thing – and different physical things at different times. As PF101 is a logical object, surely there is no reason why it should not carry on existing. But if it does carry on existing, what does this mean? Does it keep its characteristics (such as weight and RPM) from before the deinstallation? The answer is not clear.

It is also unclear when PF101 starts to exist. Some engineers thought that it started the moment the design engineer wrote it into the specification. But this account faces the same problems of explaining the pumps' characteristics when there is no physical counterpart. It also seems to confuse the existence of a concept (or sign) for something with that thing's existence. An analogy with physical pumps makes this clearer. An individual physical pump is named (and so specified) when the factory making it includes its production on its schedule. At this stage a concept (sign) for the physical pumps exists, but not the actual physical pump.

The engineers' standard properties

It might seem intuitively attractive to make obviously physical properties, such as tangibility and visibility, only apply to physical pumps. But our intuitions are less clear about what constraints there are on the engineers' – less obviously 'physical' – standard properties.

How are the engineers' standard ways of characterising a pump to be applied to the different types of object? Should one or both of the pumps have a RPM? Or a weight? Initially it seems that we should say both, as both the manufacturer and the designer give these characteristics to their pumps. But then we have a problem. Are there two different – but the same – weight properties and two different – but the same – RPM properties? And why do they have to be the same, why can't they be different? Can the necessity for their sameness be explained by there being only one weight property shared by both pumps? And if so, what happens to this property when the physical pump is replaced?

These properties suffer from the same structural, architectural problems as the more physical properties. The absolute physical/logical distinction may help to explain the 'two things in one place' puzzle for objects such as pumps. But by legitimising two distinct things in the same place, it creates a need to explain the necessary sameness of many of their basic properties – which it does not satisfy. And if would be helpful to know whether the properties were merely the same in some relevant respects or actually identical – something a criteria for identity of properties could help us establish.

3.2.2 Co-extensive things' activities

As mentioned earlier, it is not just properties that need to be explained. The activities that co-extensive physical and logical things participate in also need a framework. This has been done for some activities – install, deinstall, construct and dismantle have a framework within which the roles of physical and logical pumps are explained. But these are ones where the activities are not 'shared'. For 'shared' activities there is no general framework – and here we come across the same structural, architectural problems that we found for properties.

One of these was particularly immediate for the EPISTLE engineers. It relates to the ubiquitous pump – and its main activity, pumping.

Designers and operators design and operate pump facilities that pump. Manufacturers and maintainers produce and maintain physical pumps that pump. So pumping does not seem to be restricted to physical or logical pumps. ECM v2.22 supports this - Activity is a different subject type from Physical and Logical Object (you can see this in *Appendix D*'s *Figure MC1–0.1*) – and not subject to any physical/logical distinction[16].

But this suffers from the same intuitive puzzle as the co-extensive pumps and their properties – and gave the engineers similar intuitive concerns. There is a need to explain why the co-extensive pumps have to participate in the same activities. There is a further question about whether they are participating in two similar activities – or one. When a pump facility is pumping, the physical pump installed is necessarily also pumping. Are there two pumping activities or one? (Note that a criterion of identity for activities would help us answer this.) There is also the conceptual problem of explaining how the pump facility, which is classified as logical, can undertake a pumping activity that appears indubitably physical.

[16] Though activities may not be characterised as physical or logical, as mentioned earlier, ECM v2.22 does implement some specific constraints on whether (and how) physical and logical things can participate in some types of activity – e.g. only a physical thing can be installed by an install activity.

There is also the twist that some activities, such as install and deinstall, happen to both the physical and logical pump, but affect them in different ways. A deinstall may only involve disconnecting a physical pump – but 'destroys' the physical existence of the logical pump facility.

3.2.3 Co-extensive things' parts

As well as the co-extensive physical and logical things' relations to other types of thing (properties and activities), there are its relations to the same two types of things – in particular, the physical and logical things that are its parts.

The physical pump clearly has things such as nuts, bolts, pistons, and so on as parts – and these seem physical. But does the logical pump have parts? It would seem that this is possible, as the designers of oil rigs sometimes specify the components of the pump. And it would seem that these parts are logical. And these physical and logical parts would seem to be co-extensive – as their wholes are.

ECM v2.22 takes this view. It stipulates that parts have to be of the same entity type as the whole. The parts of a logical pump are also logical – the parts of a physical pump, physical. It is impossible for logical and physical objects to share parts. Taking this to its 'logical' conclusion, we can imagine a parallel logical universe, exactly mimicking its physical counterpart, part for part, atom for atom.

While the parallel logical universe may seem counter-intuitively over-populated, less simple strategies, bound by ECM v2.22's overall framework, lead to the need for more sophisticated and complex solutions. For example, if one allows physical and logical things to share parts – presumably physical parts – one needs to explain when and how a collection of physical pump parts compose a logical pump (and, presumably, that they can only compose one logical pump at a time). The answer cannot be 'always' as the physical parts of a manufactured pump do not always compose a pump facility. In particular, we need to explain why they do when they are installed in a pump facility.

3.2.4 Similar puzzles, suggesting a common solution

The engineers intuitive concerns were motivated, in part, by a desire for a reasonably complete and consistent story about what is going on. Where an explanation of one area (objects), was consistent with what happened in other areas (properties, activities and parts). ECM v2.22's lack of a completely convincing story shows how difficult it is to find one – the EPISTLE engineers tried very hard

Hopefully, the analysis is also beginning to show the interconnectedness of our conceptual frameworks. How adopting a structure in one area influences related areas. How adopting a structure that allows two things to be in the same place at the same time, creates a need for an explanation on how to relate their properties, activities and parts.

The similarity of the puzzles in the different areas and their structural, architectural nature suggests the possibility of a single overall solution. This is the goal of the BORO re-engineering.

4 Re-engineering the domain

4.1 Re-engineering the individuals

The purpose of BORO's re-engineering is to provide the engineers with a better – in the sense of more intuitive as well as simpler and more fruitful - story about what is going on. This, as we shall see in this section, starts with a better story of what the individuals are.

4.1.1 Grounding the analysis

It is important that the analysis has as strong a grounding as possible. The BORO categories differentiate the types of thing by the way they are grounded, and so provide a good basis for ordering the analysis. Individuals, which are directly grounded are analysed first; then the tuples that are grounded in individuals; and finally classes, which are grounded in individuals, tuples (and classes).

In each case, we take the criterion of identity, which characterises the grounding, as our starting point for the re-engineering. BORO has such a criterion, but ECM v2.22 does not provide us with much of one – as explained below, particularly for individuals. So the analysis is more about determining how to apply this criterion then re-interpreting EPISTLE's.

ECM v2.22's criterion of identity for individuals

Discussions with the EPISTLE team and a careful reading of *EPISTLE Framework V2.22* seems to show that it has no explicit identity criterion for individuals (in its terms, instances). It has something much closer to an identity criterion for classes.

For example, class and individual are described in *EPISTLE Framework V2.22's* Section 5.1 – *Instantiation* as follows:

> "Associated with an entity type is a qualifier which tells us whether an entity of that type is describing an instance of some thing or the type of thing it is.
>
> Class - a set of its members. A class is defined by a set of criteria that determine, for any thing, whether that thing is or is not a member of a class. For example centrifugal pump is a class because it is concerned with a type of equipment rather than an individual instance of that type.
>
> Anything that is not a class is an instance:-
>
> Specific - an individual occurrence of something, for example a pump with a specific serial number.
>
> … ['Typical' left out, as not relevant here]"

How do we tell whether two descriptions of a (specific) instance are of the same instance or different ones? This passage clearly has no criterion of identity to help us. The description of classes is better, but it is insufficiently clear to provide a criterion of identity. It mentions a set of criteria, but does rule on whether different sets of criteria necessarily lead to different classes? For example, if you have a class criterion of ' triangles having three equal sides' and another of 'triangles having three equal angles' do these define the same class or not? (It can be shown mathematically that all triangles meeting either one of the criteria also meet the other.) *EPISTLE Framework V2.22* does not tell us.

It would be unfair to 'criticise' ECM v2.22, as it is an example of good or even best practice in this area. Most 'standards' for modelling the world do not have as clear and rigorous a framework as it does. If current practice were for standards to try and provide us with these kinds of criteria, then one standard where one would expect to find an example *ISO 1087, Terminology – Vocabulary*. However, it defines 'object' (its name for individual) as 'Any part of the conceivable or perceivable world.' This is, from a practical point of view, just as vague as ECM v2.22.

This vagueness is not an isolated example in ISO 1087. Its name for general terms is 'concept' which it defines as 'A unit of thought constituted through abstraction on the basis of characteristics common to a set of objects.' This is, of course, a useful definition, but it does not give us enough to develop a clear criterion of identity.

There is what might seem like a practical justification for this vagueness. That it enables different projects to fit their own interpretations within the 'same' framework. However, from our perspective, this can be more of a problem than a benefit.

There is an underlying trade-off. The looser the 'definition' the wider the interpretation the different projects using the framework will make, the less likely the projects are to fit together. The tighter the 'definition' the more likely they are to fit together. Looseness may increase applicability in individual projects (though not necessarily usefulness) but it decreases the potential for interoperability. So if you are trying to provide a general framework for interoperability, for computer-based data exchange and integration - as BORO, EPISTLE and ISO 1087 are – looseness is not what you want.

As this case study will illustrate, a vital component of the 'definition' of a type of thing is an identity criterion – particularly for individuals, which ground our ontology. It may be thought that we have an instinctive intuition of what individuals are. Again the case study will show that the intuition is loose and inconsistent – so unreliable.

BORO's criterion of identity for individuals

Within BORO individuals are spatio-temporal extensions. The strategy of taking spatio-temporal extensions as individuals is well described elsewhere (see *Business Objects: Re-engineering for re-use* or *The ontology of physical objects: four dimensional hunks of matter*). However a brief recap here might be useful.

Everyone is used to thinking of bodies as extended in space. Simplifying things a little, we think of a pump as a piece of metal (matter) extended in three dimensions (maybe more if you are a sub-atomic physicist). The choice of dimensions is conventional but could be; up-down, right-left and back-forward. And we tend to think of the piece of metal as being in a particular place at each particular time (that it 'exists').

The problem with only focussing on the spatial extension is that we have to explain how something (some body) can be the same at different times – even when it looks totally different - such as, when a caterpillar turns into butterfly, or a tadpole into a frog. Adding a temporal dimension to the bodies' spatial ones side-steps this and a number of other puzzles – giving us a common basis for thinking of both bodies and events.

It also means that we have a simple criterion of identity for individuals – two names refer to the same individual if they refer to the same spatio-temporal extension. (For more explanation see *Business Objects: Re-engineering for re-use*.) As the example will make clear, this makes analysis much more straightforward.

4.1.2 Re-engineering the individual pumps

We start the analysis with the ECM v2.22's three pump instances: two physical pumps and a pump facility. Then in the next section we re-engineer their (dependent) associations.

In this section, the analysis takes the pump instances in turn – re-interpreting them as spatio-temporal extensions and identifying the extension's characteristics. This also begins to identify the common characteristics of the classes that they belong to.

Characterising a spatio-temporal extension

An individual's principle of identity is its spatio-temporal extension. So when we characterise this extension, we are characterising its identity. We have found that, in practice, the most effective way to do this is to describe its relationships with other individuals in the domain – that is other extensions, focussing the following four key types of relationship (which all have a mereological or topological character) – where an object:

- is a part or contains as a part another individual (mereology),
- overlaps another individual (mereology – the individuals contain a common part) and
- is spatio-temporally connected in some way to another individual (topology).

When an individual does not have a continuous connected existence – either in time or space, it is useful:

- to characterise the object's continuous (maximally) connected parts and their relationships to one another (mereology and topology).

Another particularly useful technique is considering where and when an object's extension begins and ends in time – again in relation to other extensions. As was pointed out in the 17th Century by John Locke (in *An Essay Concerning Human Understanding*[17]) if two things have different beginnings (or endings) they cannot be the same thing.

[17] Book II, Chapter xxvii, 1 – XXVII – Of identity and diversity – "… When we see any thing to be in any place in any instant of time, we are sure, (be it what it will) that it is that very thing, and not another, which at that same time exists in another place, how like and indistinguishable soever it may be in all other respects: … For we never finding, nor conceiving it possible, that two things of the same kind should exist in the same place at the same time, we rightly conclude, that whatever exists any where at any time, excludes all of the same kind, and is there it self alone. When therefore we demand, whether any thing be the same or no, it refers always to something that existed such a time in such a place, which 'twas certain, at that instant, was the same with it self and no other: From whence it follows that one thing cannot have two beginnings of Existence, nor two things one beginning, it being impossible for two things of the same kind, to be or exist in the same instant, in the very same place; or one and the same thing in different places. That therefore that had one beginning is the same thing, and that which had a different beginning in time and place from that, in not the same but divers. That which has made the Difficulty from this Relation, has been the little care and attention used in having precise Notions of the things to which it is attributed."

Normally the analysis focuses on the relationships between the individuals in the selected domain. However, there are cases where an individual's extension has important mereo-topological relationships with individuals outside the domain. In this case, we naturally extend the domain. In this case study, there is an example of this in the relationship between the pump facility and the oil rig in which it is installed.

We illustrate the results of the analysis using a diagramming technique called space-time maps, which was developed to clearly represent these relationships.

The physical pumps

We start by analysing the key characteristics of the spatio-temporal extensions of the two physical pumps, XYZ1234 and XYZ5678. This is easy – as they are quite simple. We start by considering where and when their extensions begin and end in time.

Both pumps were physically constructed at some time and place. Pump XYZ1234 was constructed at factory X at time T_1, pump XYZ5678 also at factory X, but at time T_2. The particular details are not so important for this example (the factories are outside the domain), just the principle that their construction is easily identifiable.

The pumps have existed continuously in both space and time since then. (For simplicity, we presume that the pumps have not been comprehensively dismantled and reassembled – this would raise unrelated issues.) And their extensions have not overlapped. We can diagram the characteristics of these extensions as simple boxes on a space-time map – as shown in *Figure MC1–7* below.

In BORO, all individuals are physical (there is no physical/logical distinction), so calling the pumps 'physical' is tautological. Even though it may cause some initial difficulties it is better in the long run if we use the more descriptive name, 'constructed pumps'. This is to contrast them with pump facilities that, as we shall see later, are ontically dependent upon the plant of which they are components. For the same reason, we also rename 'pump facilities' to 'component pumps'[18].

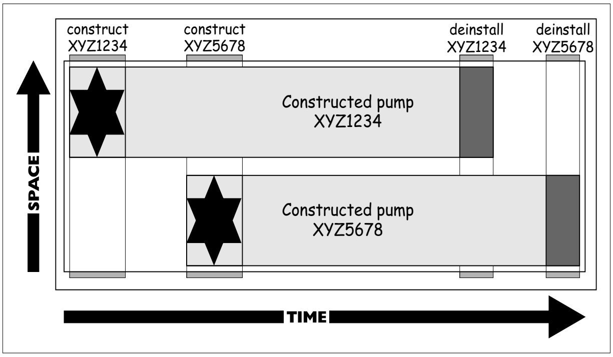


Figure MC1–7 Constructed pumps' space-time map

The component pump (facility)

If we add the component pump (facility) PF101 to the analysis, the picture is not quite so simple – not only is there the need to characterise its relationships with the two physical pumps, but also to (following John Locke's 'technique') identify its beginning and end.

To do this, we need to add some more context to our example. We clarify that constructed pump XYZ1234 was the first pump installed in the component pump PF101 during the construction of the oil rig and constructed pump XYZ5678 is the only replacement in its whole life – deinstalled during the dismantling of the oil rig.

Then, it seems obvious to me that PF101's beginning is the installation of XYZ1234 – and that once XYZ1234 was installed, they had the same spatial extension. After all, before the installation we could not see or touch PF101 – and after it, we could. Applying a similar 'see and touch' test to the end of PF101 tells us that this happens when and where XYZ5678 is deinstalled, during the dismantling of oil rig.

This means we can now characterise PF101. It starts (when and where) XYZ1234 is initially installed in the oil rig. From that moment, until XYZ1234's deinstalla-

[18] Some EPISTLE engineers have been considering 'material' and 'functional' as the names for the two types of the distinction. I do not find these attractive. Both constructed and the component pumps are material – and not every breakdown of an oil rig into components has to be into functional parts: e.g. a breakdown by floor level. Eventually the analysis will reveal a cleaner constructed - dependent distinction – generalising component into dependent. However, while considering pumps the more specific 'component' with its sense of part seems a better level to work at.

tion, it shares the same spatial extension. This means that XYZ1234 and PF101 share a temporal part (a temporal part is a time-segment of an object or, saying the same thing in a different way, the full spatial extent of an object for a period of time). PF101 then 'continues' as a temporal part of the new constructed pump XYZ5678, one that also lasts from its installation to deinstallation. This means that PF101 ends when and where XYZ5678 is deinstalled.

Note that this means (loosely speaking) that the pump PF101 ceases to exist between the deinstallation of XYZ1234 and the installation of XYZ5678 (and this is the way BORO answers the engineers' concerns about 'logical existence' mentioned earlier in *§.What is logical existence?*). More accurately it is in two temporally disconnected parts. This notion of a disconnected thing seems counter-intuitive to some people. However, while disconnected things are not the most typical of material things, they are not that unusual. For example, when we take a tent down, we are dismantling it into separate (spatial) pieces. Similarly a football match, which is played in two halves, is also disconnected (like PF101) in time, but does not seem counter-intuitive – until, maybe, we focus on its disconnectness.

These characteristics are diagrammed in the space-time maps in *Figure MC1–8* below. The first map is from PF101's perspective – showing PF101. The second map is from XYZ1234 and XYZ5678's perspective, showing how their temporal parts compose PF101.

Figure MC1–8
Component
pump PF101's
space-time map

In general, we say that a component pu

seem sensible to assume that different 'ontologies' will at least agree on an item's basic category – who would mistake a body for a relation?) Then all we need to do is translate them into BORO's version of a relation (a tuple of the things it associates which, in turn, belongs to a corresponding relation class). This is tempting as it is extremely simple to translate the two installation ECM v2.22 associations into two BORO relationships; one between PF101 and XYZ1234, the other between PF101 and XYZ5678.

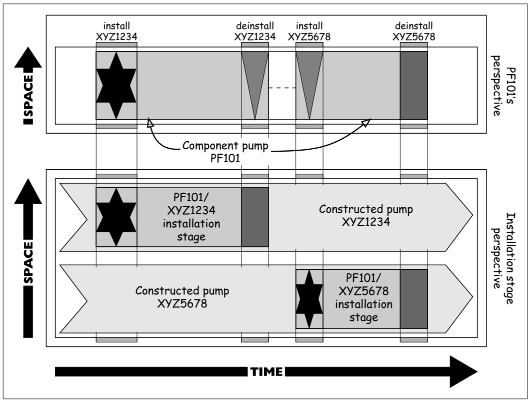
The nature of the installation associations

But this simple translation will not work, as this thought experiment shows. In BORO a relationship is a tuple that belongs to a relation class. If we were to analyse the ECM v2.22 installation associations as relationships, then we would say that they were the two couples - <PF101, XYZ1234> and <PF101, XYZ5678> - both belonging to a relation class, installations. But if we now change the example slightly we can see this will not work.

Consider the situation in which we have a third installation – where XYZ5678 is replaced with XYZ1234, the original pump. We know this is a new installation, but

PF101/XYZ5678) are visible in *Figure MC1–8*, but not marked as objects. This is remedied in *Figure MC1–9* below.

Figure MC1–9 PF101's installation stages space-time map



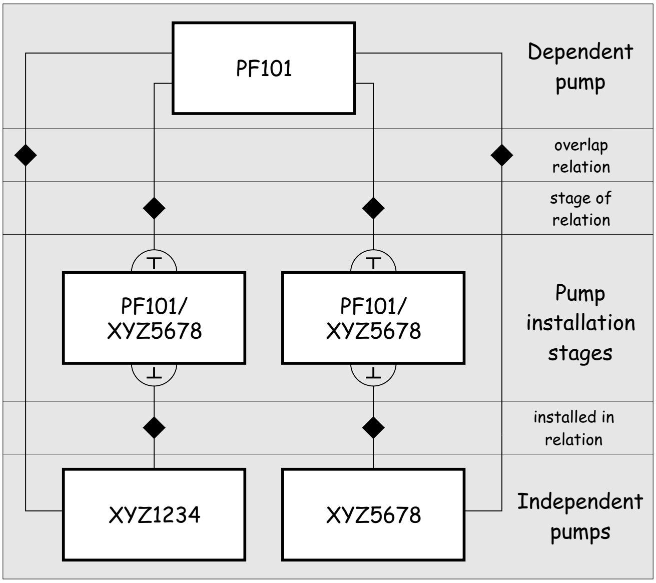
These temporal parts are also the stages in PF101's life. Just as people have childhood and adulthood stages (and lepidopters have caterpillar and butterfly stages), PF101 has (and component pumps, in general, can have) installation stages.

Not just any temporal part of PF101 is an installation stage, only those that mark the lifespan of the installation of a physical pump. The start and end of each installation stage is marked by its install and deinstall activities – which we look at in the next section. The full extension of each installation stage is crisply marked out by the overlap of its wholes. *Figure MC1–9* shows this - for example that the PF101/XYZ1234 installation stage is the overlap of the 'wholes' PF101 and XYZ1234. In general, the extension of the installation stage is the overlap of the extensions of the component pump and the constructed pump installed in it. This overlapping is explicitly shown in *Figure MC1–10* below. Where only one constructed pump can be installed at a time, this gives us the pattern exemplified by PF101's life stages – where one, and only one, stage follows the other.

n ECM v2.22 an installable link joined the installation association to the physical pump and a service link joined it to the pump facility. The re-engineering has revealed these links as structural – as temporal parts. In BORO we rename the installable link 'installed in' and the service link 'stage of' – as shown in *Figure MC1–10*.

People who have not been exposed to much ontological analysis often think that commonsense makes it intuitively obvious whether something is an individual or a relation. But, as this analysis illustrates, it is not always true. The EPISTLE team saw installation as an association, whereas BORO reveals it as an individual. Experience with BORO analysis confirms that relying on intuition to identify an object's basic category can be risky. The ontological framework provides important clues to what things are (according to the framework), which mitigates this risk – for example, the clues this analysis relied on were the criteria of identity for individuals and tuples.

Figure MC1–10 Installation stages object schema



4.1.4 Re-engineering the installation activities

Given that the re-engineering of installation associations revealed them as installation stages, what does this imply for their installation activities? In ECM v2.22 these are the activities that begin and end the installation associations[19].

Maybe in BORO the activities can be seen as marking the beginning and end of the installation stages.

But to be sure about this we need to consider what the installation activities actually are. In the space-time map in *Figure MC1–9* above, the install activity was idealised into an instantaneous event involving only the pumps. Now that we are looking more closely at the activity, we need to fill in more details of the picture.

The install XYZ1234/PF101 activity (see *§PF101's installation associations' install and deinstall activities*) involves the maintenance engineer and tools (such as spanners) as well as the pumps. But it only involves these for a period of time around when the pump is being installed. And different things are involved for different periods of time – for example, the spanner may only be used in the sub-activity of bolting the pump down. In general, the precise spatial extension of an activity is more difficult to tie down than its precise temporal extension – whereas with physical things, such as pumps, the opposite is usually true.

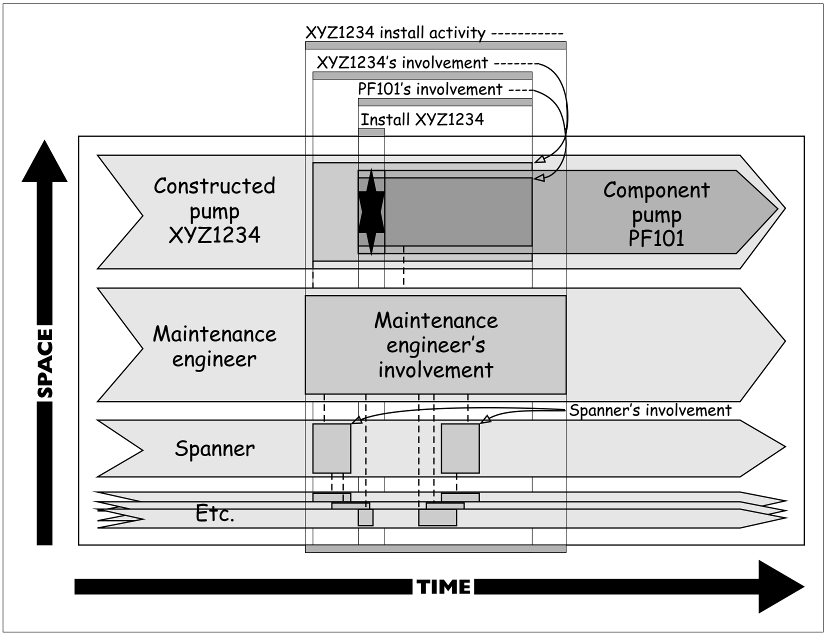
We can consider the activity as the fusion (sum) of the things involved in the activities for the period of time that they are involved in the activity. This means the install activity has as parts a number of temporal parts of things of different types; for example, a maintenance engineer temporal part, a spanner temporal part, as well as pump temporal parts.

Let's assume that we have a simple case where all the maintenance engineer does is pick up the pump, place it in position, use a spanner to bolt it down and then make the various connections. It is not clear in this description what pump we are talking about (constructed or component), so let's make this clear. It seems to make sense to say XYZ1234 starts becoming involved in the activity when the engineer picks it up. But it also seems reasonable to say that PF101 does not become involved (indeed does not exist, in one sense) until XYZ1234 is in position. One might want to refine the details to be more exact about the involvements, but the picture is clear enough to reveal that the two pumps have different tem-

[19] "Conceptually, the start and end points of the lifetime of an association are defined by the activities that bring the association about and terminate it." §4.3 Association, *EPISTLE Framework V2.22*.

poral parts involved in the activity: PF101's involvement and XYZ1234's involvement. Where PF101's involvement is a part - indeed, a temporal part - of XYZ1234's involvement. These situations are shown in *Figure MC1–11* below – note that this has been simplified to only identify the main things involved in the install activity. The figure also neatly illustrates how an event's spatial extent is determined by the spatial extent of its participants.

Figure MC1–11 PF101–XYZ1234's installation activities space-time map



PF101 is a discontinuous (in time) individual with the two temporal parts (installation stages) of the *same* kind of thing – pumps. Here, unlike PF101, we have a number of *different* kinds of things whose temporal parts make up a single individual. And unlike, PF101 these parts are typically connected somehow. The maintenance engineer picks up the spanner, the spanner is used to tighten the pump's nuts, etc. Drawing this stretches the limits of the space-time map's ability to represent, so I have resorted to dashed lines to show where the things touch.

A similar situation happens at the deinstallation. Assuming again that it is a simple sequence of sub-activities; where all the maintenance engineer does is disconnect the pump by, say, undoing some bolts, and then take it out of its position. It seems to make sense to say that both XYZ1234 and PF101 start becoming involved when the engineer starts making the disconnections. It also seems reasonable to say that PF101 stops being involved (indeed stops 'existing', in one sense) when XYZ1234 is taken from its position. This again implies two different pump temporal parts are involved in the activity: PF101's involvement and XYZ1234's involvement. And again, PF101's involvement is a part - indeed, a temporal part - of XYZ1234's involvement.

The PF101/XYZ1234 deinstallation activity provides us with a good example of how temporarily (temporally) overlapping things develop different histories – how they start to, and stop, overlapping. When the maintenance engineer picks up XYZ1234 and (re)moves it from its position in the oil rig, PF101 does not move – it just ceases to 'exist' (in one sense). Seeing and touching XYZ1234 (as described earlier) does not distinguish between PF101 and XYZ1234, but the removal – as part of a deinstall activity – does, it only applies to XYZ1234.

These objects are modelled in *Figure MC1–12* below, which shows both PF101's install and deinstall activities.

Figure MC1–12 – PF101's installation activities - structural - object schema

Figure MC1–12 focuses on the structure of the activity. For example, it shows PF101's involvement as an individual – and reveals the structural nature of the relationship between PF101 and the activity, they share a part. Also, to keep the schema simple, we have shown PF101's involvement via the PF101/XYZ1234 installation stage.

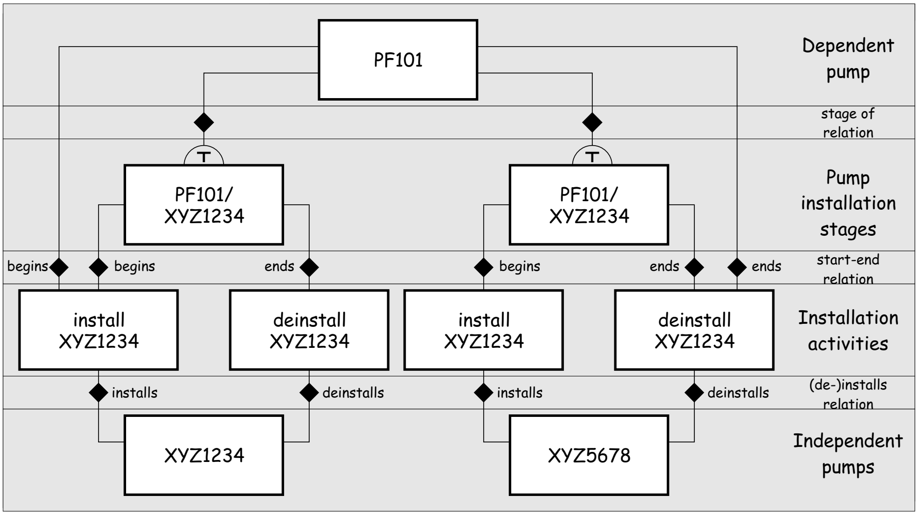
However, from a practical point of view, it is more useful to think of PF101/ XYZ1234, PF101 and XYZ1234 as directly involved in the activity via an involve-

involvement relation also marks the start and end of the installation stage and the component pump.

Beginning and ending is an important aspect, so it makes sense to follow ECM v2.22's intuition that activities mark the beginning and ending of associations and extend this to other relevant things. However ECM v2.22 stipulates each association has a different begin and end activity. BORO's criterion for identity helps us to recognise that a more flexible structure is required in its wider application; that, for example, the same install activity starts both PF101 and PF101/XYZ1234 (and the same activity ends both).

It is also worth noting that install (or deinstall) activities do not always start (or end) the same types of thing. An initial install, such as install XYZ1234, marks the start of a component pump and an installation stage. Whereas each subsequent install, such as install XYZ5678, only marks the start of an installation stage. This revised perspective is shown in *Figure MC1–13* below – note how the initial install has two start relations and the subsequent install only has one.

Figure MC1–13
PF101's installation activities – start and end - object schema

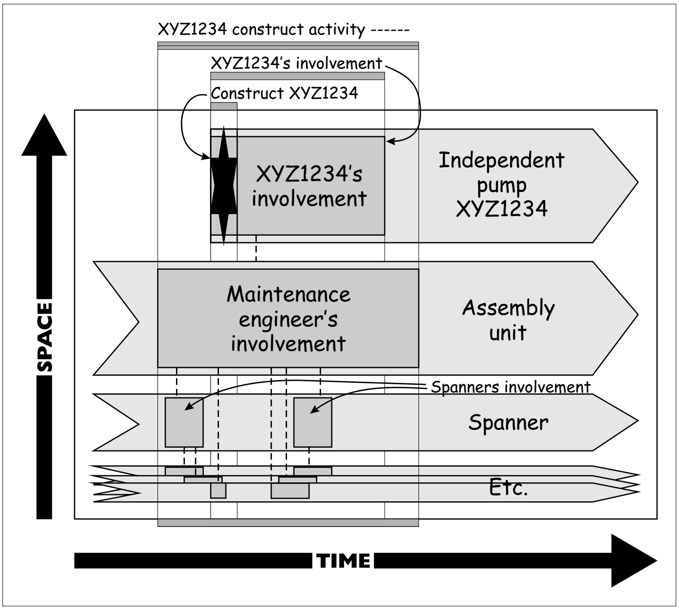


4.1.5 Re-engineering the construction activities

As the parts of XYZ1234 are assembled, as XYZ1234 is constructed, it comes into existence (begins). As the parts are disassembled, as XYZ1234 is dismantled, it goes out of existence (ends).

The nature of the construction (construct and dismantle) activities of XYZ1234 and XYZ5678 is similar to that of the installation activities. This can be seen in the similarity between its space-time map in *Figure MC1–14* below, and installation's *Figure MC1–11*.

Figure MC1–14 XYZ1234's construct activity space-time map



There are two points worth noticing. Firstly, the exact moment that we want to say, for example, XYZ1234 begins to exist – the point from which its temporal extent begins – is to some extent a matter of convention[20]. It does not seem to make a lot of difference whether we say "this happens when there is more than

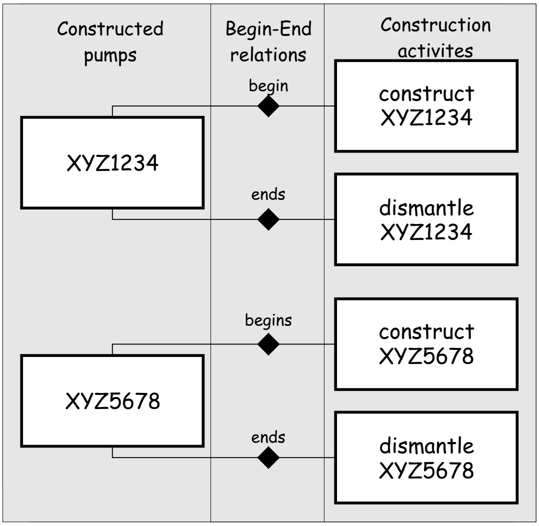
[20] Remember that we noted earlier that the precise temporal extent of bodies such as pumps was typically harder to determine than its precise spatial extent – this is a good example.

one part" – or "it only happens when all the parts are in place". The only requirement is that, when relevant, there is a convention that ensures that the members of the community using the ontology recognise a sufficiently similar point as the start.

Secondly, not any 'putting together' or 'taking apart' of any number of parts counts as a construct or dismantle activity – it has to be one that constructs or dismantles a pump. Otherwise, for example, routine maintenance where some parts of the pump were taken out would lead to the pump going out of existence. An extreme case of this is a trombone which is regularly taken apart to store in its case. This is a borderline case – but most people would want to say that it continues to exist in pieces.

The object schema in *Figure MC1–15* below shows the 'start-end' view of the construction activities for both constructed pumps – with a direct link between the participants in the activity and the activity, rather than via their temporal parts (as in the 'structural' *Figure MC1–12* above).

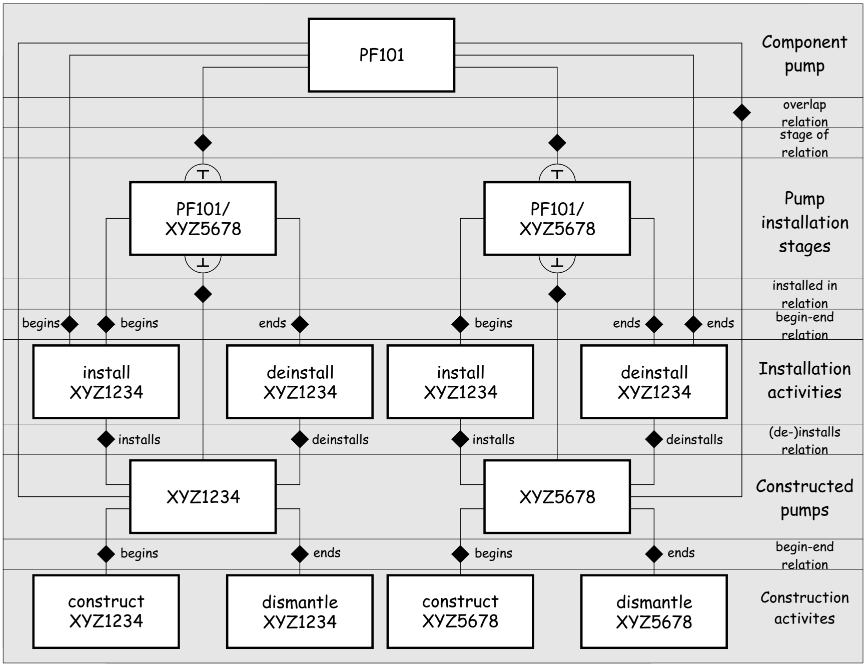
Figure MC1–15
Constructed pumps construction activities – start-end - object schema



4.1.6 BORO's new ground level ontology

We now have a decent picture of the new ground level of individuals (and their inter-relationships) – and this is diagrammed in *Figure MC1–16* below. The temporal part analysis of the activities' relationship with their participants has been left out, in favour of the more direct 'involving' relationships.

Figure MC1–16
The new ground level ontology



If you compare this with ECM v2.22's ground level ontology in *Figure MC1–6*, the first thing that strikes you is the structural similarity. The two figures have the same basic structure – the same number of components with almost the same relations. But behind this are major architectural differences that will become apparent when we re-engineer the upper level.

The ontological differences

Underneath the structural similarities are these four ontological differences:

1 BORO has is no absolute physical-logical distinction. All its objects are physical. In BORO, component pumps (facilities) are differentiated from constructed (physical)

pumps by their differing extensions and the different activities that begin and end them. This enables it to accommodate (unlike ECM v2.22) a general notion of pump – and also give a firm footing to the notion of the component pump's existence.

2 BORO allows component and constructed pumps to share parts. So what ECM v2.22 calls installation associations, BORO regards as the installation stages that overlap (and so are temporal parts) of the constructed pump and the component pump. This moves them from the ECM category of relations (associations) into that of individuals.

3 BORO's spatio-temporal approach enables it to explain the nature of ECM v2.22's (non-structural) installation associations in terms of structural (temporal-) whole-part relations.

4 The spatio-temporal approach enables BORO to generalise these to cover the different types of pumps – recognising construction and installation as beginning and ending the various types of pumps. Within ECM v2.22, activities are only used to mark the start and end of associations. Also BORO's more flexible structure accommodates the install activities starting two different types of pump, component and installation stage – something not possible in ECM's more rigid structure.

As we have noted during the analysis, these types of changes are typical of this type of re-engineering. The more precise BORO framework provokes a fundamental re-assessment within which some things reveal a completely different nature, such as the installation association – and its links (and as we shall see in later analysis, some things disappear completely).

Structural relations

The revealing of structural relationships is particularly characteristic of BORO re-engineering. In fact, in this example all the relationships between the individuals in the BORO ontology are either structural whole-part or based upon structural whole-part relationships – in the case of activities. People tend to feel that structural relationships (once they have got used to them) give a more intuitive explanation of what objects are than, for example (as the EPISTLE engineers found) ECM v2.22's install association's non-structural service and installable links.

Another aspect of this more structural explanation, which people have noticed, is that many of the extensions of individuals can be defined in terms of the extensions of other individuals. For example, PF101 can be delineated as the fusion of the two installation stages. These stages can, in turn, be delineated in terms of their component and constructed pumps. For example, PF101/XYZ1234 can be delineated as the overlap of PF101 and XYZ1234 – or as the overlap of the oil rig and XYZ1234. Similarly, PF101/XYZ5678 can be delineated as the overlap of PF101 and XYZ5678. These last two turn out to be specific examples of a reasonably general rule – that temporal parts typically can be delineated as the overlaps of objects.

4.1.7 Resolving some intuitive concerns

Even at this stage, we can begin to see how BORO's approach to the core ontological question resolves the EPISTLE engineers' intuitive concerns. We first look at how BORO deals with the core ontological question, then a pragmatic explanation of the operator and the maintainer's different views – and then at how the engineers' specific concerns about ECM v2.22's solution are resolved.

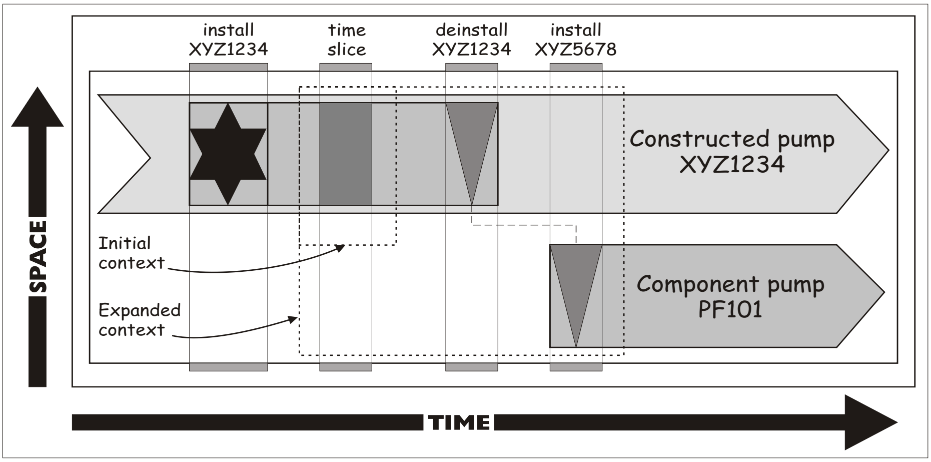
The core ontological question

And if you need to know that you come back to the ontological question – can two things be in the same place at the same time? This To answer the question it makes sense to clarify what a place is within BORO.

Places as time-slices

An object's place can be regarded as its spatial extent at that particular time – where a time might be an instant or an interval. This is probably best explained in terms of time-slices of individuals. If we think of an individual, such as XYZ1234, as extended in time – and then take a slice out of that extension at a particular time, we have a time-slice. This is shown diagrammatically in the space-time map in *Figure MC1–17* below.

Figure MC1–17 Place, time, context space-time map



And time-slices fully occupy their places[21]. So for time-slices the answer to the core ontological question is “no” – only one time-slice can be in the same place at the same time. This is guaranteed under BORO’s criterion of identity for individuals, which says that there can only ever be one individual with this spatio-and-very-brief-temporal extension.

But it is possible for two more typical objects, such as XYZ1234 and PF101, which are reasonably extended in time, to share time-slices and so share the same place at a time. This is clearly shown in *Figure MC1–17*. This means the answer to the ontological question for these kinds of objects is “yes” - at a time.

The distinction between time-slices and the ‘typical’ objects of which they are slices gives us an explanation for our intuition that two things cannot be in the same place at the same time.

While these relatively arbitrary time-slices are useful as a tool for building up our understanding, they do not seem to have a particular role in this example – so we do not include them in the ontology.

[21] I am speaking a little loosely here. Strictly speaking, in BORO time-slices are their places – as there is no separate category of location. However there is no harm done, as location is not really a pertinent point here.

It is worth pointing out that within BORO the traditional ontological question has a related spatio-temporal one:

> Can two things have exactly the same spatio-temporal extent?

In other words, can they always be in exactly the same places at exactly the same times. In BORO the answer to this is “no” – its criterion of identity for individuals says that they would be the same thing. Note that ECM v2.22 is neutral on this; it gives no reason why a logical and physical thing should not have exactly the same spatio-temporal extent.

Explaining the apparent operator and maintainer contradiction

We can expand the notion of timeslices to explain the apparent contradiction raised earlier in *§.The core ontological issue* (in *§2.3*) – where the operator and maintainer seem to be pointing at both the same and different pumps. It also highlights a fundamental feature of how we ground individuals.

To see how all this works assume that we are with the operator and maintainer and are looking at, and touching the pump at a time just after XYZ1234 is installed. We casually say we can see the pump and we can touch it. More accurately, we actually only see and touch the surface of a part for a short period of time. We cannot see the back or inside of the pump. We cannot see back into its past or forward into its future. However, within the context of the occasion we can extrapolate an extension for the pump - shown as a context box in *Figure MC1–17*.

The operator and maintainer are in a better position. They know their pumps and can fill in more of their histories. However they also appreciate that only they know this and that as far as the group is concerned the discussion is restricted to the context. When the extension of the operator’s and maintainer’s pumps are restricted to the context - one ends up with a single extension and so the pumps are within the context pragmatically the same. So it is quite in order for everyone to agree that they are talking about the (pragmatically) same pump.

By the time of the second meeting, the context has been extended to include the replacement of the pump - this is also shown as a context box in *Figure MC1–17*.

Now the pumps' restricted extensions are different. This explains why the maintainer and operator then claim that there are two pumps.

Refining an ontology from an epistemology

An ontology is intended to describe the world as it is – independently of what we know. Whereas an epistemology (in the sense we are using here) describes what we know about the world.The contexts we have just described are a feature of an epistemology. They reflect what the group knows - rather than a way the world is.

The task of the ontological analysis is to refine the epistemic elements to reveal the ontology behind. From an epistemic point of view, we can say the operator and maintainer's pumps are pragmatically the same in the first context - but they are most definitely ontically different.

The EPISTLE engineers' intuitive concerns

With the BORO 'explanation' in mind we can resolve the EPISTLE engineers' intuitive discomforts with their framework. In *§3.2*, we started with simple commonsense questions, such as:

> When the physical pump XYZ1234 is pumping, what is the logical facility PF101 doing?

Or even more basically:

> When I touch physical pump XYZ1234, am I also touching logical facility PF101?

We identified the concerns as coming from an inadequate characterisation of the architectural (ontological, even) framework for the relationships between physical and logical objects occupying the same place (i.e. co-extensive) and:

- the properties they have,
- the activities they participate in, and
- the parts they have.

In general, these problems disappear in BORO along with the absolute physical-logical distinction.

In the earlier section, we examined how the physical properties of tangibility and visibility caused particular concern. How could the necessary equality of logical PF101's and physical XYZ1234's spatial extension, visibility and tangibility (while XYZ1234 was installed) be explained? The problem arises because, in ECM v2.22, there are two different things both present in different ways (physically and logically) at a particular time.

The problem does not arise in BORO where the only 'thing' that is completely there at the particular time is the time-slice discussed in the previous section. And there is only one of these to see and feel. And this one thing – this time-slice – is a temporal part of both PF101 and XYZ1234, which both have a spatio-and-longer-temporal extension.

Other properties, including the engineers' standard characteristics for a pump (such as weight and RPM) succumb to a similar treatment. The properties at a time – in other words, the properties of the time-slice – are the same because there is only one thing - the time-slice. The properties also belong to the temporally extended individuals, but these can differ to extent. A particular constructed (physical) pump is likely to have a relatively stable weight and RPM – at least in theory. So we can easily think of characterising it with a (reasonably) specific weight. However, the different constructed pumps installed in a component pump (facility) may have differing weights and RPM's, but ones that vary within a permissible range. So we can characterise these in terms of allowable ranges. When we measure the weight or RPM at a particular time, then we are measuring a time-slice's properties, which is a temporal part of the constructed and component pumps.

A similar thing happens to activities. It is a time-slice of the pumps that engage in an activities – such as pumping. And there is only one such time-slice. Things are a bit more sophisticated with activities in which the co-extensive pumps participate in different ways – such as install. Here different temporal extents of the pumps participate – and with different results.

Again, a similar thing happens to the pumps' parts. As, in BORO, both types of pumps are physical, they can share parts simpliciter. However, we need to distinguish between ‘real’ spatio-temporal parts – such as the component pump is a part of the oil rig – and merely temporally overlapping parts – such as the constructed pump being (a spatial) ‘part’ of the oil rig for a particular time.

All in all, providing a sufficiently sophisticated strategy upfront for dealing with the core ontological question means that there are less intuitive discomforts down the line.

4.2 Re-engineering the upper ontology

The focus when re-engineering the individuals is on providing as firm as possible a grounding. When re-engineering the upper ontology, there is the additional need focus shifts towards building a coherent and consistent system – on how the various objects fit together, from the top-level categories down to the individuals. This typically begins by making explicit the informal assumptions used in the analysis of the individuals. A key tool for this is classes – and a key part of the analysis is grounding the class by characterising its extension.

4.2.1 Characterising classes’ extensions

Within BORO a class’s extension is the basis for its principle of identity. So we can characterise its identity by characterising its extension (just as we did for individuals). Unlike individuals, classes’ extensions are not directly grounded - they consist of its members – and the members’ extensions, if individuals, are directly grounded in their spatio-temporal extensions. We have found that, in practice, the characterisation of the classes works at both these levels.

At the class level, we try and determine whether a class:

- is included in or includes another class (i.e. all the member of one class are also members of other class).
- overlaps another class (i.e. the classes have some, but not all, common members), and

- is disjoint from another class (i.e. the classes have no common members). Where a number of classes are mutually disjoint, this is characterised in terms of a partition.

A common manoeuvre is to regard the class inclusion relation (also known as the super-sub-class relation) as a kind of whole-part relation over classes – then the process above can be seen as determining the mereological characteristics of the class (in a similar way to that done for individuals – see *§.Characterising a spatio-temporal extension* in *§4.1.2* above).

By the time we come to an analysis of a class, we will usually have analysed some individual members of the class – and so have some idea of their mereo-topological characteristics and whether these are common to all members of the class. This provides us with basis for doing a class level characterisation of the individual members mere-topoological characteristics. In other words, whether all the individual members of a class:

- are parts of or contain as a part individual members of another class (mereology),
- overlap the individual members of another class (mereology), and
- are spatio-temporally connected in some way to the individual members of another class (topology).

This in no way exhausts the ways in which the class can be characterised, but it does highlight what are usually the important ways of characterising it – as the case study will illustrate.

4.2.2 Re-engineering the pump class

In BORO the ‘mechanism’ for classifying is membership of classes. When re-engineering the individuals, they were informally named as constructed or component pumps. We now formally model the ontology of these classifications. The obvious starting place for our analysis is pumps. As we shall see BORO’s more flexible framework allows a less awkward classification of pumps than ECM v2.22.

An inflexibility in ECM v2.22's framework

ECM v2.22's treatment of pumps classes is hampered by a rigidity in its framework. This has its roots in the following three elements, which, taken together, lead to an awkward constraint:

- The strategy of having physical things and (logical) facilities as different entity sub-types of the subject dimension.
- The decision to have an instantiation dimension containing class as a subtype.
- The rule that - an entity instance "can only can only be placed in precisely one entity type in each of the four dimensions".

You can see the first two elements clearly in ECM v2.22's framework structure diagrammed in *Appendix D*'s *Figure MC1–0.1*.

These three elements taken together ensure that no class (entity instance of the class sub-type) can be an instance of both the physical and logical things subtypes – as they belong to the same dimension, subject.

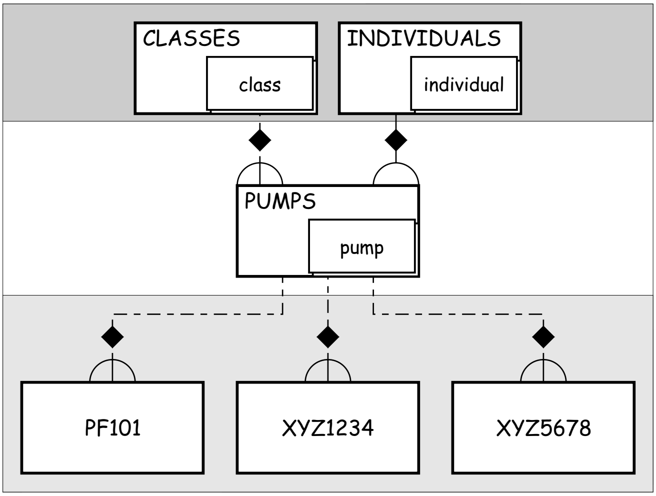
In particular, this means that ECM v2.22 cannot have a single pump class to characterise pumps, it has to have two (the physical pump and pump facility class – shown in *Figure MC1–1* and *Figure MC1–2*). This is intuitively unnatural. Dictionaries have only one entry for things like pumps – it is not unreasonable to expect the ontological framework to be allowed to do the same. The EPISTLE engineers using the ECM v2.22 made the same point.

BORO's general pump class

BORO is not constrained in the same way. Within BORO's framework it makes sense to have a general pump class – for pump's common characteristics[22]. As this is a class, it is a member of the BORO framework class, classes. As it contains only individuals, it is a sub-class of the individuals class. PF101, XYZ1234 and XYZ5678 are all members of it. The model for this is in *Figure MC1–18* below.

22. [2] One reason is that it makes class a major category in its ontological framework, rather than an entity subtype. The ontological shift that takes 'class' and 'instance/individual' from a separate ECM v2.22 dimension to BORO categories is substantial and is going to be the subject of a separate case study. So we do not explore it any more here.

Figure MC1–18
General pump class object schema



4.2.3 Analysing the life-history ‘dimension’

The key ECM v2.22 distinction vis a vis pumps (and other equipment) is an absolute one between the physical and logical facility subtypes on the subject dimension. This analysis reveals how the distinction – now between constructed and component – is reflected in different life histories. Later analysis will deal with component pumps’ ontological dependence upon their oil rigs.

Pumps’ life history ‘dimension’

The model in *Figure MC1–18* is incomplete. It just shows pumps, not constructed and component pumps – something clearly shown in the ECM model. Before we can model this, we need to better understand what differentiates them.

We can perhaps start to get a better understanding of the constructed/component distinction by contrasting it with a different kind of distinction: the distinction between a centrifugal and a screw pump. These two distinctions are orthogonal. A centrifugal and a screw pump can be either constructed or component; or put the other way around, both a constructed and a component pump can be either a centrifugal or screw pump. It is clear what differentiates a centrifugal and a screw pump, the way in which they work – using a centrifuge or a screw to pump. But what differentiates a constructed and a component pump (and, it turns out, an installation stage)?

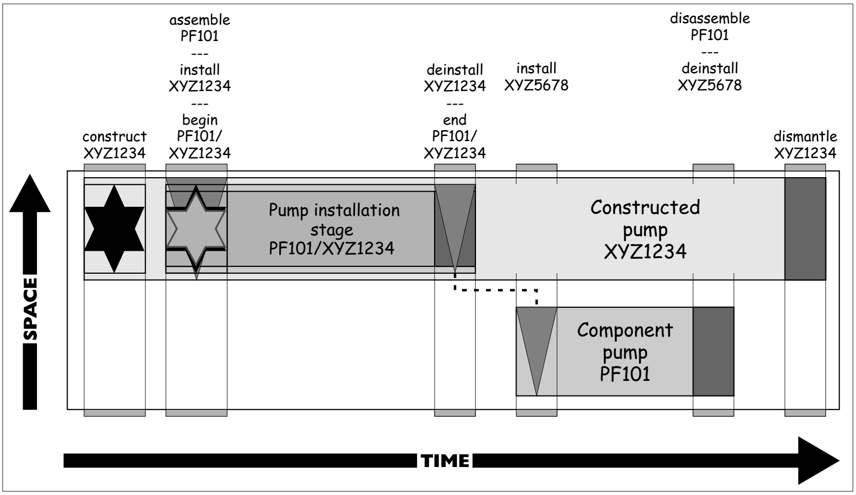
An important difference is the pattern of its life history[23]. Perhaps this is easier to see with an illustration. Assume XYZ1234 has just been installed in PF101. The constructed XYZ1234, the component PF101 and the installation stage PF101/ XYZ1234 all overlap. In other words, at that time they all have the same spatial-extension. So they must share the same spatial characteristics, for example, the same spatial parts. It seems that what these pumps – including the pump installation stage - have in common is their spatial characteristics. This suggests, quite correctly, that what differentiates them is their temporal characteristics. When we consider this we notice that the constructed pump has a different life history from the component pump and the installation stage. They begin and end at different times – and their beginning and ending is (usually) the result of different events.

Conversely, a constructed centrifugal pump and a constructed screw pump would have different types of spatial extension, but similarly shaped (in the relevant respects) temporal extensions – similar life histories. For example, they would both be constructed – unlike a component pump, which would be installed.

This life history pattern is fruitful and, as we shall see later, it has a wide range of applications. A space-time map showing examples of each of these life history types is given in *Figure MC1–19* below.

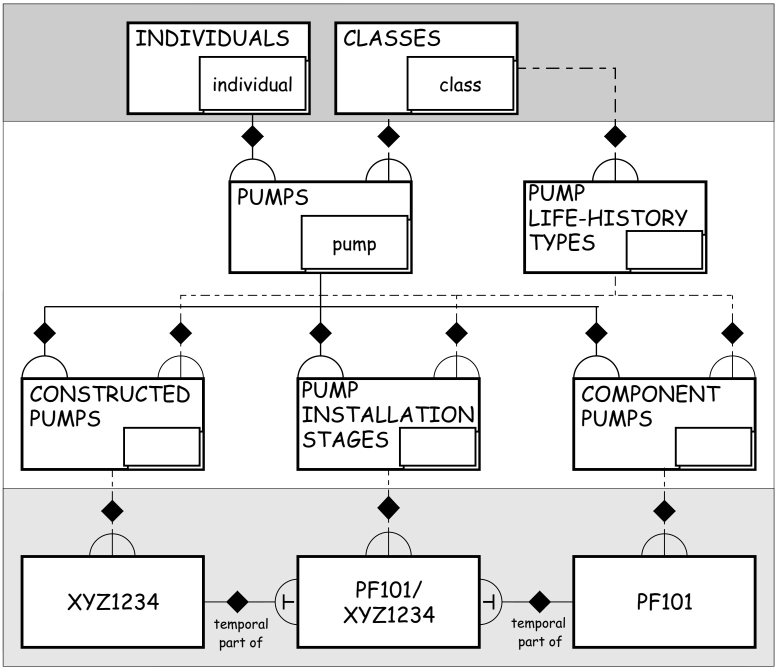
[23] Another important, and reasonably obvious, difference is that component pumps have an ontic dependency upon their whole, whereas constructed pumps do not. As ECM v2.22 does not regard this relation as important, it is currently outside scope – we introduce it in a later section.

Figure MC1–19
Pump life history types space-time map



For a different perspective on the same objects, the general pump class and its life history subclasses are shown in *Figure MC1–20* below.

Figure MC1–20
Pump life history sub-classes object schema



Also included in *Figure MC1–20* above is the class (of classes) – pump life history types. This raises an important point in development of ontological frameworks. In ECM v2.22 the dimensions, such as subject, are talked about – rather than considered as things (in philosophy-speak, reified) and included in the ontology. From a framework point of view, it would be better if these were explicitly recognised. In BORO we do this using a class of classes – here the 'pump life history types' class that has classes, such as component pumps, as members.[24]

A quick word here on terminology. Traditionally engineers use the name *equipment* for physical 'equipment' and the name *tag* for 'equipment' facilities. This is too cumbersome for our purposes here and does not highlight the similarities between the two types of thing. So we use 'constructed equipment' for the traditional 'equipment', 'component equipment' for the traditional 'tag' and 'equipment' as a general term covering both constructed and component equipment.

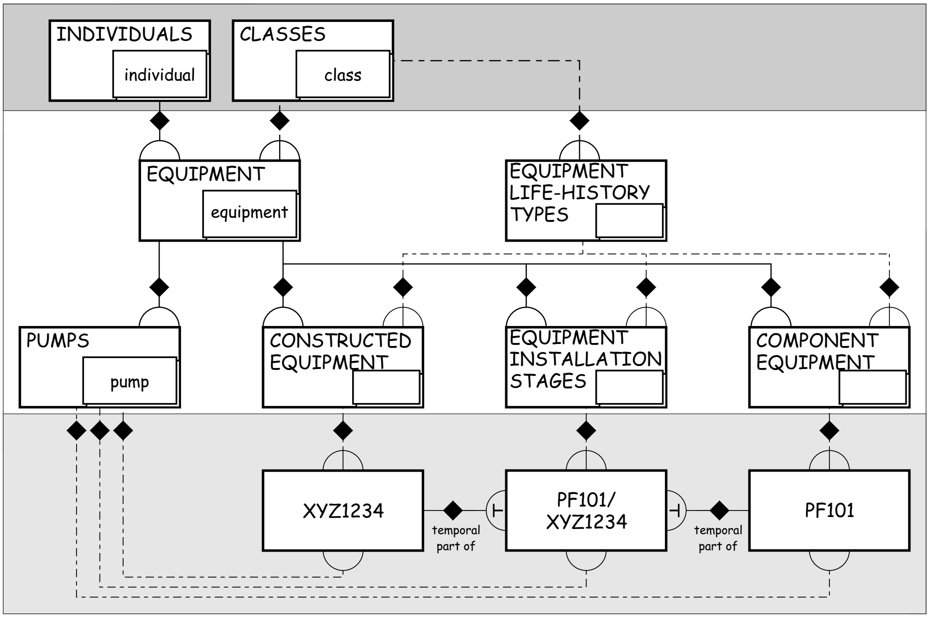
Generalising to equipment

The introduction of the life history types class provides us with a clue that there is an opportunity for generalisation. It prompts the question of whether these types apply exclusively to pumps – and if not, how generally this pattern applies.

It is plain that pump PF101 is not the only piece of component equipment in the oil rig, all its other 'tags' in the traditional sense (things such as valves and air conditioning units) are as well and these have their constructed counterparts. There is every reason to suppose that BORO's analysis of component and constructed pumps in terms of life histories would equally apply to these. A model showing the result of doing this is in *Figure MC1–21* below.

[24] This general point about reifying – not just talking about - important classifications is going to be covered in more detail in another case study, so it is not discussed any more here.

Figure MC1–21 General equipment class object schema

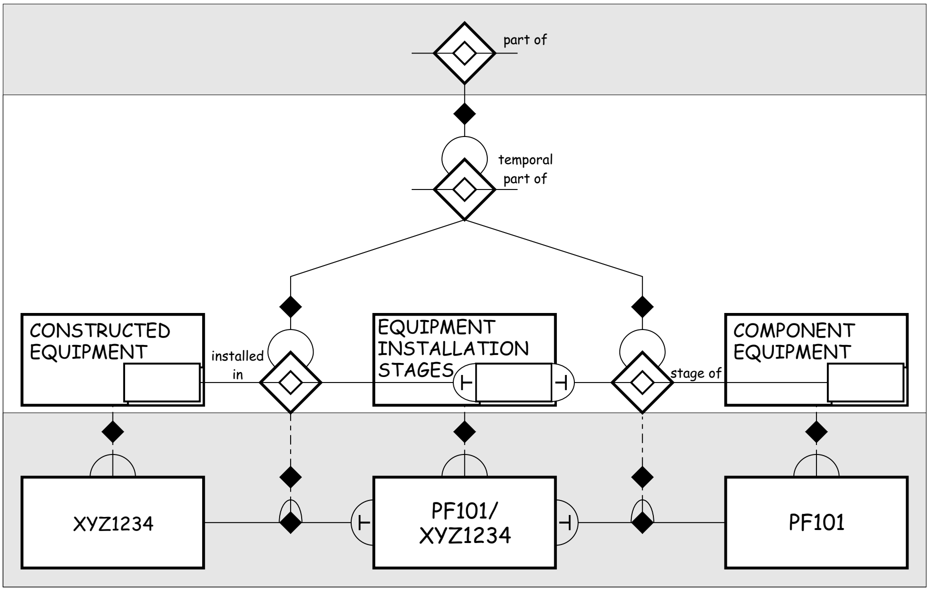


On the face of it, there seems no reason why the life history types pattern cannot reasonably be raised to the level of equipment. We now illustrate how this generalisation might work. There is insufficient spcae here, for the full analysis that would 'justify' this.

4.2.4 Re-engineering the install associations' relations

The analysis that led to *Figure MC1–21* above focussed up the class-member hierarchy on the life history types class. We now turn our attention to the relations between the members of this class - to the installation stages relations with the constructed and component equipment: the 'installed in' and 'stage of' relations. We can generalise these relations to the equipment level – as shown in *Figure MC1–22* below.

Figure MC1–22 Installation stage relations object schema



4.2.5 Re-engineering activities

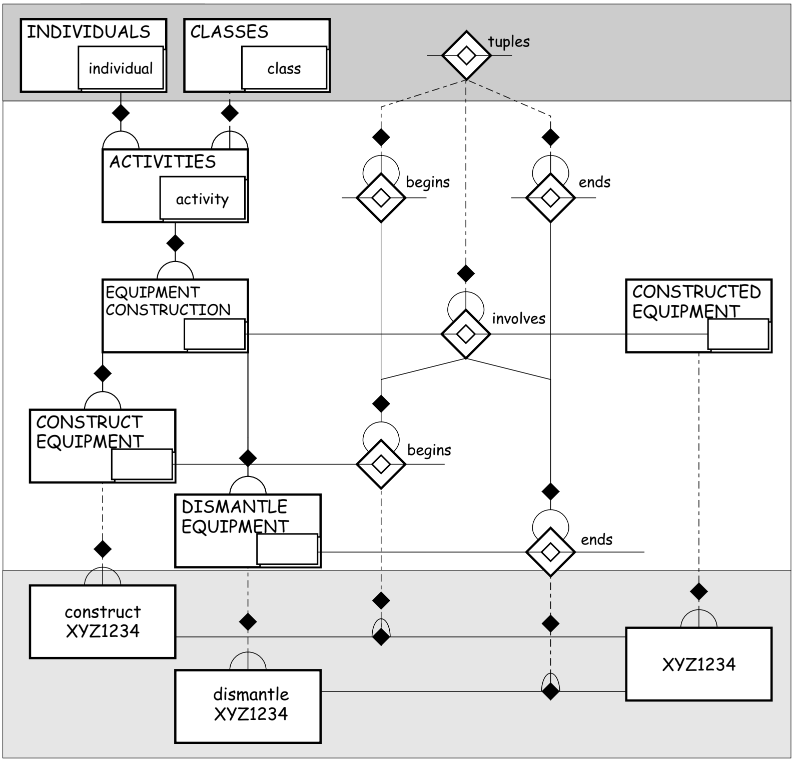
The ECM model identifies two types of activities to re-engineer:

- Construction activities, and
- Installation activities.

Re-engineering construction activities

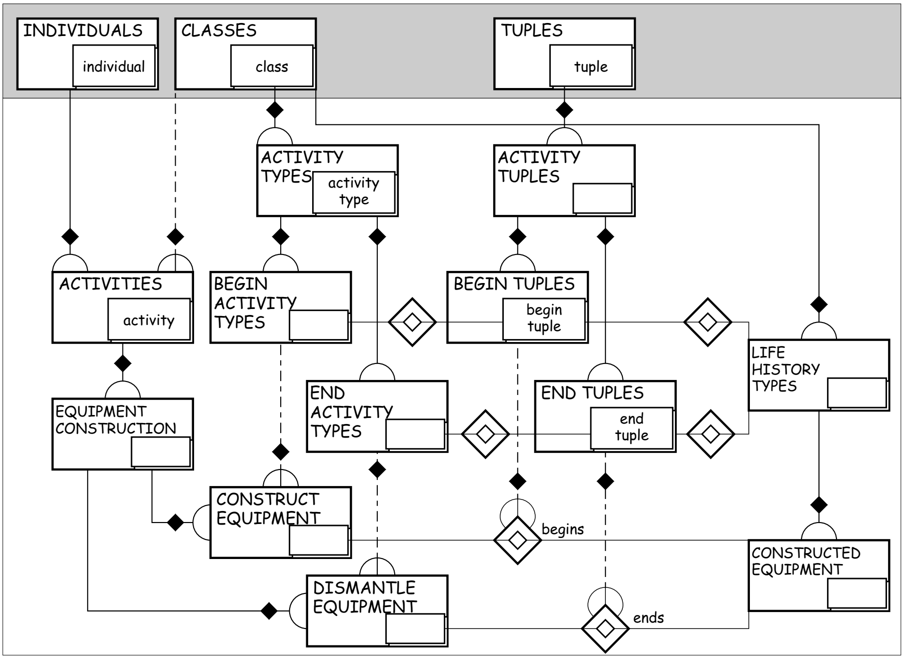
As in the ECM model, we can take the individual construct and dismantle activities as belonging to construct and dismantle activity classes. Within the BORO framework we have the flexibility to have an equipment construction activities class, whose instances always involve constructed equipment, with construct and dismantle sub-classes. And if the construction activity is a construct activity, the involvement starts the equipment's life – and if it is a dismantle activity, it ends the equipment's life. This is shown below in *Figure MC1–23*.

Figure MC1–23 Equipment construction activities object schema



Just as the different types of life histories were 'encapsulated' by a life history types class – here the begin and end relations can be 'encapsulated' by begin and end activity and relation types. This opens the door to a higher level of generality, where we say that begin and end activity types begin and end life history types. And then we can be more specific and say which activity types begin and end which life history types. For example, that the construct activity begins and the dismantle activity ends constructed equipment. The result of saying this is modelled in *Figure MC1–24* below.

Figure MC1–24 Equipment begin and end activity types object schema



Re-engineering installation activities

This encapsulation/generalisation naturally leads us towards an analysis of the types of activities that begin and end the various life history types. And installation activities are an obvious place to start – as they seem to have a similar underlying pattern. Install activities always begin installation stages and deinstall activities always end them.

However, as noted in *§4.1.4* and shown in *Figure MC1–13*, the overall picture is not quite as simple here. Two life history types are involved – component equipment and installation stages – and not every installation activity involves both life history types. We have here a pattern where every install and deinstall activity marks the beginning and ending of an installation stage. But only the initial install marks the beginning and the final deinstall activity marks the ending of the component equipment.

4.2.6 Characterising life history types with activities

What differentiates the life history types are their life histories. Particularly the type of their begin and end activities. We now look at how we can capture this insight (though again there is insufficient space for a full analysis). We recognise a relationship between life history types and begin and end activity types, where each life history type has its own begin and end activity type – as shown in *Table MC1–1* below.

Table MC1–1 Life history activities

Life history type	Activity type	Activity
Constructed equipment	Begin	Construct
	End	Dismantle
Component equipment	Begin	(Initial) install
	Begin gap	Non-final deinstall
	End gap	Subsequent install
	End	(Final) deinstall
Equipment installation stage	Begin	Install
	End	Deinstall

4.2.7 Ontology versus epistemology, again

Table MC1–1 implies that every instance of equipment has a begin and end activity that marks its beginning and ending. And that each life history type has its own type of begin and end activity. This seems to make sense, as one of the things that could differentiate things with different life histories is different types of beginnings and endings – things with the same types of beginnings and endings could be further classified by the types of events that shaped their life histories.

This also provides us with another good example of the difference between an ontology and an epistemology. From an ontological point of view, every piece of equipment has a begin activity. Whereas, from an epistemological point of view, a system may not know (or need to know) what a particular piece of equipment's begin activity is. For example, we know that physical pump XYZ1234 has a begin activity, but the maintenance engineer and the maintenance systems, would typically not know what it was.

4.2.8 Slimming down the ontology

So far in this re-engineering of the upper level, we have been adding to the original ECM v2.22 ontology (for example, we have added the pump, equipment and equipment life history classes). One of the key features of the BORO approach is using generalisation to slim down the ontology. We got a glimpse of how this works in the way in which the 'equipment life history types' subsumed the 'pump life history types'. We get a better look at the way BORO encourages slimming down when we re-engineer the extended ontology in the next section.

5 Extending the ECM v2.22 domain

In ECM v2.22 the key facility relationship is the installation association, which shapes the relationship between physical equipment and facilities. BORO recognises this relationship, but views another relationship, component of, as more important in characterising components (facilities).

So far we have only informally mentioned this ontically 'dependent' aspect – noting that the component pump is dependent upon the oil rig of which it is a component.

Within the ECM v2.22 framework the component relationship between oil rig and pump facility is not seen as critical and so not naturally included in the application model (in fact, as we shall see, it is difficult to model properly within ECM v2.22). Within BORO this relationship characterises the component pump's *dependency* – so is critical. We need to extend the scope of the domain to include this relationship – in other words, include the oil rig and its relations with the pumps.

5.1 Extending the ECM v2.22 application model

We start by extending the ECM v2.22 application model. While working with the ECM v2.22 model we revert where appropriate to ECM v2.22 names (e.g. pump facility rather than component pump).

5.1.1 ECM v2.22's unwritten convention

ECM v2.22 has an unwritten convention that is pertinent to our analysis. This convention evolved out of lessons learnt by the engineers applying ECM v2.22's physical/facility distinction and its associated installation associations. It divides the kinds of engineering artefacts that the Process Industry deals with into three levels based (roughly) upon size and applies different rules for the use of the distinction.

- The 'big' level contains mostly plant and its sub-units. Things such as offshore oil-drilling rigs and ships and their sub units. A conscious policy on only realising these as facilities is adopted. So there are no 'big' physical plant and 'big' facilities are never physicalised by installation in a physical plant.
- The 'medium' level contains mostly equipment. Things that are attached by nuts and bolts . Equipment comes in two varieties: constructed-physical and component-facility. Where component-facility equipment tends to be described in specifications as components of things at the

'big' level and identified by tag nos. And the constructed-physical equipment is installed in the component-facility equipment.

- The 'small' level contains mostly what can be called component-parts[25]. These are typically parts of things at the 'medium' level, things such as pistons and wiring – or the nuts and bolts to hold them in place. These are normally only realised as constructed-physical objects and used to 'assemble' constructed-physical things at the medium level. There are exceptions, such as three inch diameter bolts costing GBP20k each, for which there are good cost control reasons for introducing their component-facility counterparts.

There are good practical reasons for this practice, but in terms of the ECM v2.22 framework, it is a workaround. Ships and offshore oil-drilling rigs are plainly physical – if anything is – rather than logical facilities. For our analysis we need to see the nature of the ECM v2.22 framework, so we do not use the workaround, but apply the framework strictly. This will reveal counter-intuitive aspects that need re-engineering.

5.2 ECM v2.22's practical approach

However, we start by building an application model of the workaround approach that regards the oil rig as only a facility. We know that pump facility PF101 is a component of an oil rig. The component part relation is realised in ECM v2.22 as an assembly association between the pump facility and the oil rig, which is a subtype of the composition association - ECM v2.22's name for the whole-part relation.

The assembly association forces us to recognise that the oil rig is a facility, because ECM v2.22 restricts the part of relation to things of the same generic subtype. As the pump facility is a facility so its whole, the oil rig, must also be a facility.

The motivation behind ECM v2.22's strategy is sensible. It makes clear simple distinctions and enables the more general use of the physical-install-facility pat-

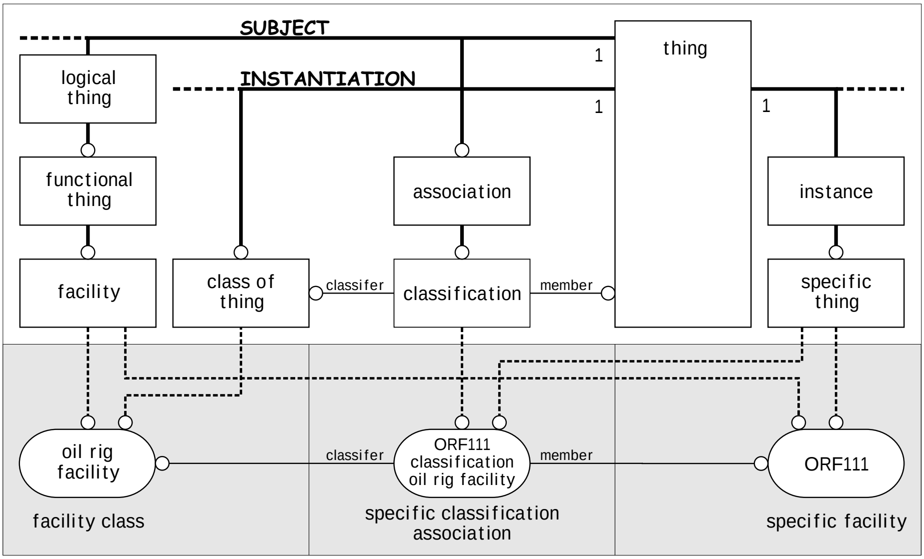
[25] It would be better to have a word other than 'component' as a component of the name – as it already has a use. But nothing better has suggested itself yet.

tern (enshrined in the installation association). However, in this case it looks as though the strategy leads to intuitively awkward results.

5.2.1 ECM v2.22's conventional oil rig facility

ECM v2.22 conventionally only recognises one oil rig - a (logical) oil rig facility (tag number ORF111, say). This logical oil rig facility is fitted into the ECM v2.22 framework with a class and classification association, in an analogous way to the pump facility (see *Figure MC1–1*) – as shown in *Figure MC1–25* below.

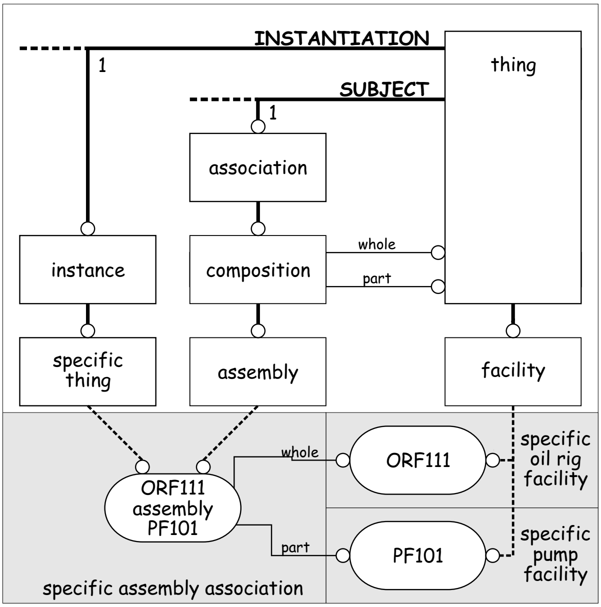
Figure MC1–25
Oil rig facility



5.2.2 The oil rigs' conventional assembly association

Pump facility PF101 is linked to the oil rig facility ORF111 by a specific instance of a new type of association, *assembly* – where the assembly association is a subtype of composition, ECM v2.22's whole-part relation. This new type of association diagrammed in *Figure MC1–26* below.

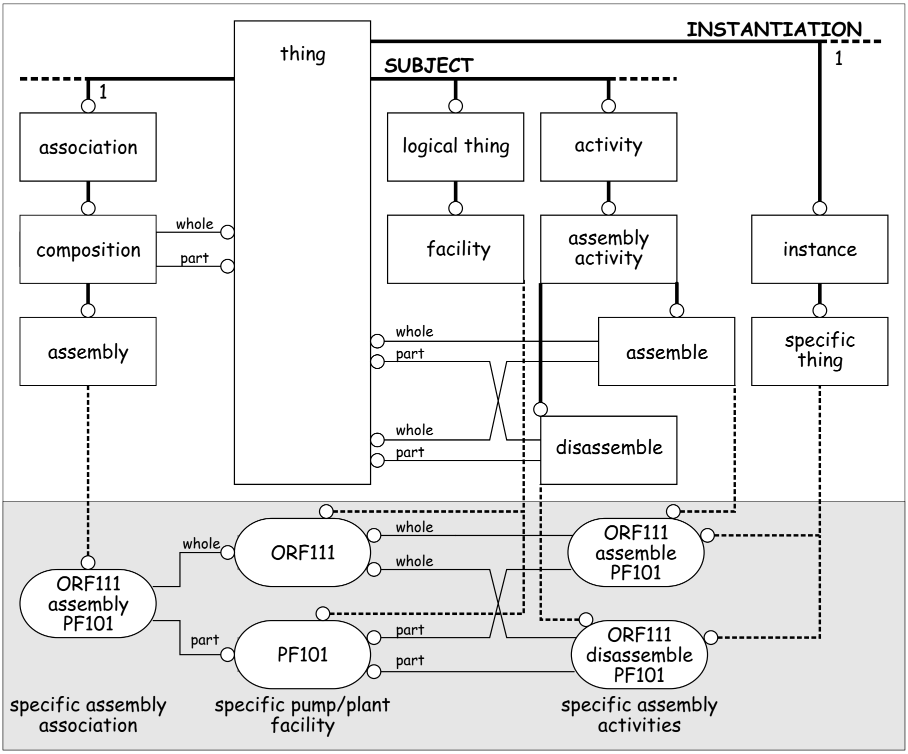
Figure MC1–26
Oil rig-pump-facility specific assembly association



5.2.3 The conventional associations' activities

The assembly association has an assemble activity that marks its beginning and a disassemble activity that marks its ending. These are shown in *Figure MC1–27* below.

Figure MC1–27
Oil rig-pump-facility assembly activities



However it is not clear what these activities are. When an association has a lifetime then it seems reasonable to have activities that mark its birth and death. This works admirably for installation associations where there is some physical process to mark the activity. However, things are less clear with this assemble association, probably because it is between logical things – so it cannot really have a physical assemble activity. And, as mentioned earlier in §3.2.3, ECM v2.22 is unclear about the existence criteria for logical objects, so we are not even sure when these come into and go out of existence. An apparent candidate is the initial installation and final deinstallation of the physical pumps – but these are installation not assemble activities. One suggestion is that the assembly association starts when the requirement for PF101 as a part of ORF111 is added to the specification. But this implies PF101 and ORF111 also exist at that time. It also sheds no light on when or where all these logical things could end. The BORO analysis clarifies the situation.

5.3 A strict(er) ECM v2.22 approach

As mentioned earlier, to get a full understanding of the nature of the ECM v2.22 framework we want to see the ontology that results from applying a strict approach – without workarounds.

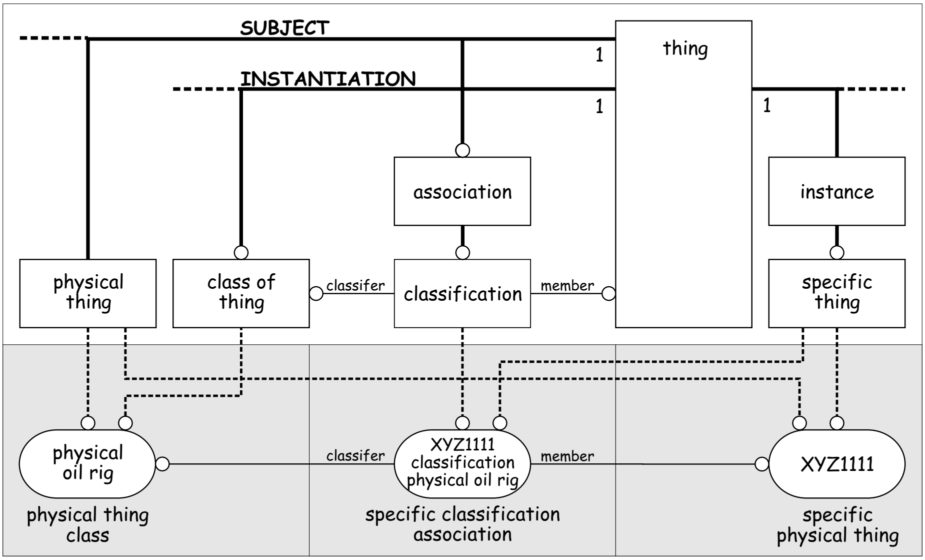
This involves two steps:

> Firstly, it forces us into the awkward position of having a surplus of oil rigs. Of having to recognise a physical oil rig as well as an oil rig facility. The oil rig facility is a logical thing – whereas it is plain to everyone who comes into contact with an oil rig that it has a physical presence. So we 'physicalise' it (as we 'physicalised' the logical pump facility) by installing a *physical* oil rig.
>
> Secondly, once we have the physical oil rig, we have to explain its links to the physical pumps. We do this within the ECM v2.22 framework using the assembly association.

5.3.1 A strict physical oil rig

First we recognise a physical oil rig which we call XYZ1111 – oil rigs are usually too big to have serial numbers. *Figure MC1–28* below shows how the physical oil rig is fitted into the ECM v2.22 framework – in the same way as the physical pumps were (see *Figure MC1–2*).

Figure MC1–28
Physical oil rig



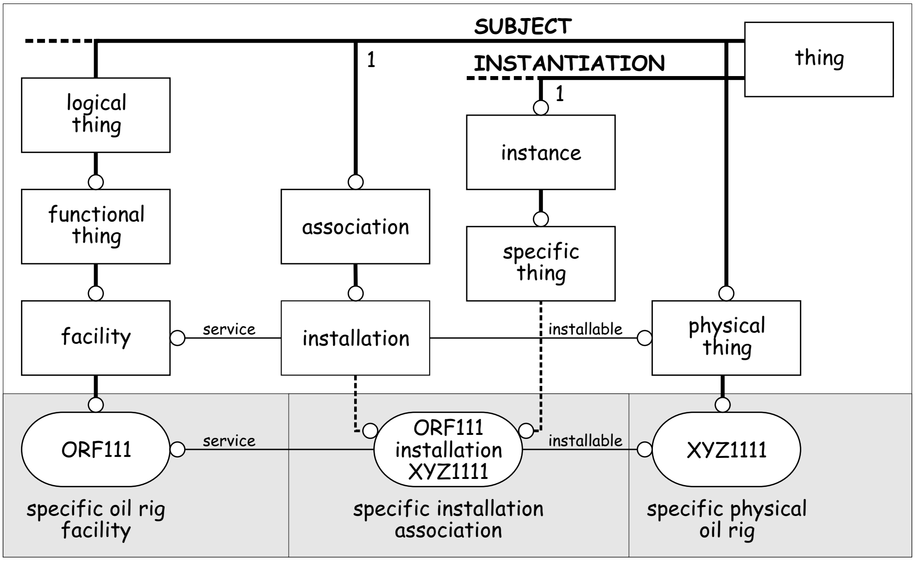
As noted earlier for pumps, the ECM v2.22 framework ‘forces’ you to have separate physical and logical classes for the different types of oil rig. As with pumps, the BORO analysis removes this counter-intuitive restriction.

5.3.2 The physical oil rig’s specific installation association

By analogy with the physical pumps, physical oil rig XYZ1111 has a specific *installation* association with oil rig facility ORF111, analogous to the physical pumps’ installation associations with pump facility PF101.

The specific installation association is diagrammed in *Figure MC1–29* below.

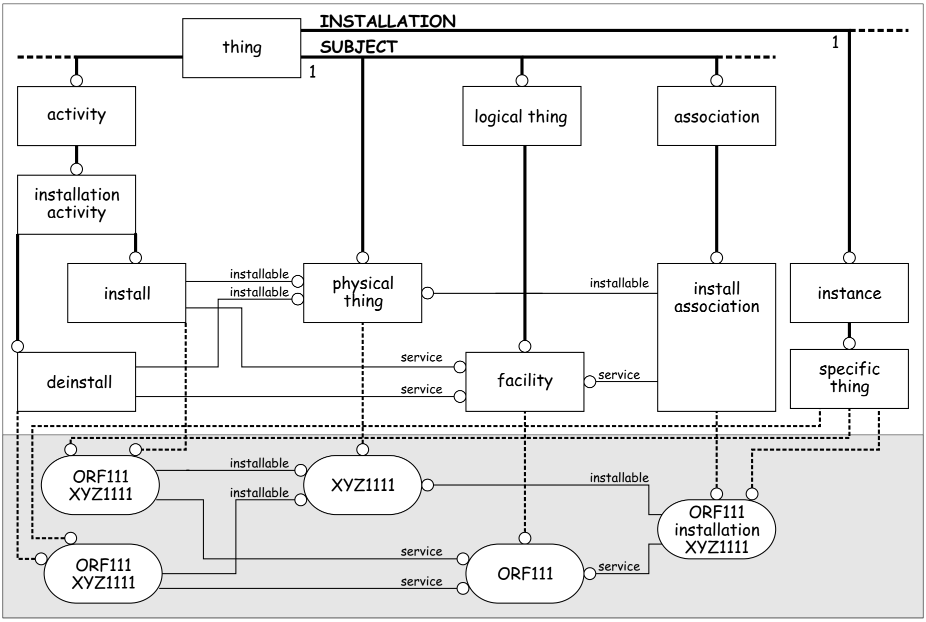
Figure MC1–29
Oil rig specific installation associations



5.4 The installation association's activities

We now recognise the physical oil rig's installation associations' activities. Just as PF101's installation started and ended with activities – so does ORF111's installation association with XYZ1111. It starts with the activity of installing XYZ1111 and ends with the activity of deinstalling XYZ11111. It is easy to model this – see *Figure MC1–30* below – but it is less clear what this actually is.

Figure MC1–30
Oil rig facility ORF111's installation activities

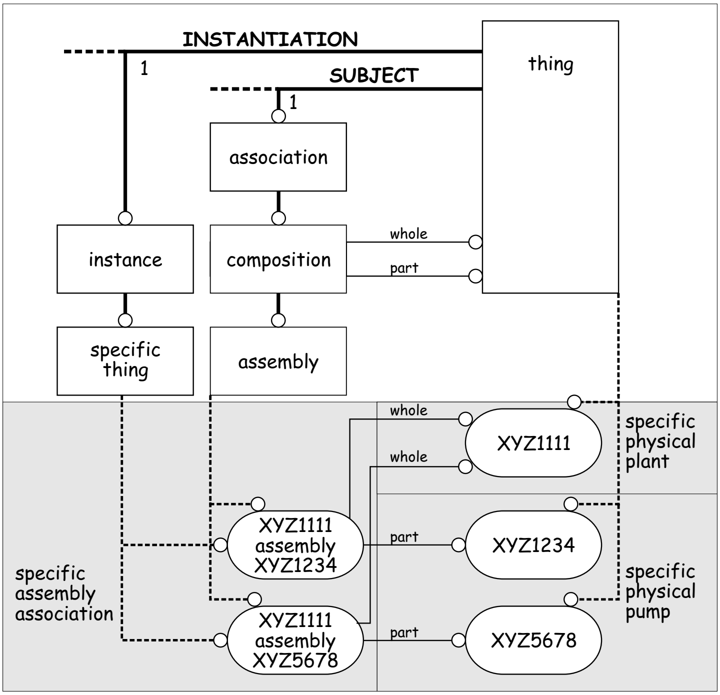


We have not really considered what these activities could be. As with ORF111-PF101's assembly activities above (in *§5.2.3*), it is difficult to identify what these could be as they do not seem to correspond to any physical process. We do not pursue this point as the association disappears in the BORO re-engineering.

5.5 The physical oil rig's construction activities

For completeness, we follow the approach taken for physical pumps and specify the beginning and end of physical oil rig, XYZ111; in other words, its construction activities. These are shown in *Figure MC1–31* below.

Figure MC1–31
Physical oil rig XYZ111's construction activities



5.6 The physical oil rig's relationship with the physical pumps

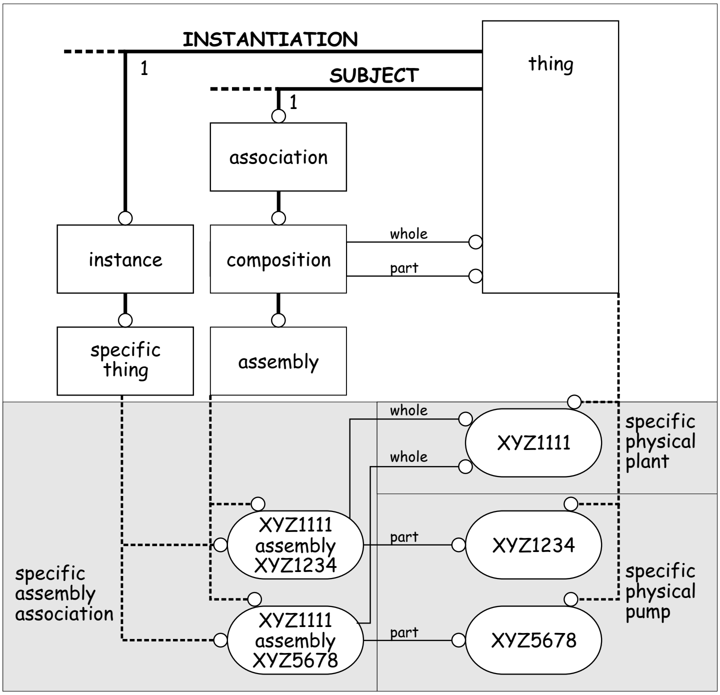
It is clear that once we have introduced the 'big' level physical oil rig XYZ1111 into the ontology, we need to understand its relationship with the physical pumps. However conventional ECM v2.22 ontologies usually avoid having physical things at the 'big' level so we are, to an extent, in relatively unexplored territory here.

When we look at the physical oil rig and its pumps there seems to be a direct 'part of' relationship – as well as the less obvious indirect relationship (via the installation association with the pump facilities and its assembly association with the oil rig facility and its installation association with the physical oil rig).

There is what seems like an analogous situation in conventional ECM v2.22 at the 'medium' level of pumps. The physical pumps' physical components are related by assembly associations. When these components are added to the pump and replaced, this is realised in the ontology using an assembly association and its assemble and disassemble activities.

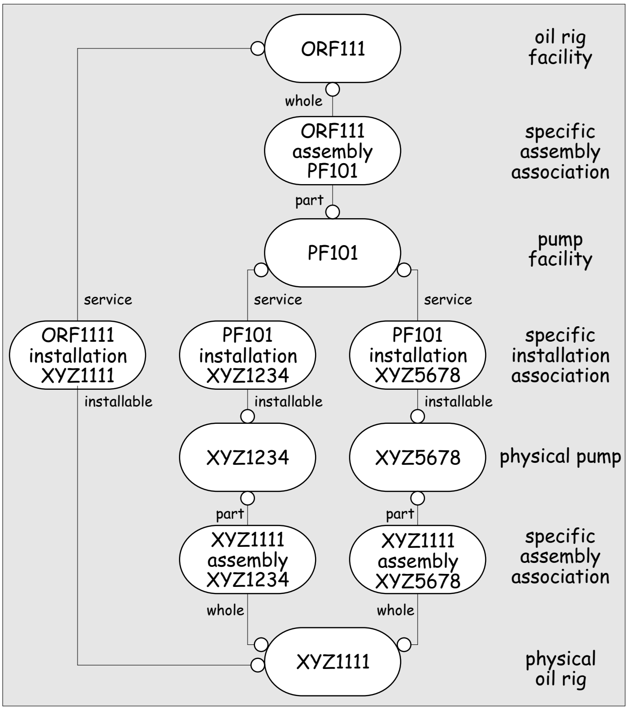
We take our lead from this analogy and recognise XYZ1111's specific assembly associations with XYZ1234 and XYZ5678 shown in the model in *Figure MC1–32* below.

Figure MC1–32
XYZ1111's assembly associations



The 'beginning-ending' principle then compels us to recognise XYZ1111's two assembly associations' assemble and disassemble activities - modelled in *Figure MC1–33* below.

Figure MC1–34 The extended example's ground level ECM v2.22 ontology



The eighteen activities are shown in the table below.

Table MC1–2 ECM v2.22 Activities

Association	Activity
XYZ1234 installation PF101	XYZ1234 install PF101
	XYZ1234 deinstall PF101
XYZ5678 installation PF101	XYZ5678 install PF101
	XYZ5678 deinstall PF101
XYZ1111 installation ORF111	XYZ1111 install PF111
	XYZ1111 deinstall PF111
PF101 assembly ORF1111	PF101 assemble ORF1111
	PF101 disassemble ORF1111
XYZ1234 assembly XYZ1111	XYZ1234 assemble XYZ1111

Table MC1–2 ECM v2.22 Activities (continued)

Association	Activity
	XYZ1234 disassemble XYZ1111
XYZ5678 assembly XYZ1111	XYZ5678 assemble XYZ1111
	XYZ5678 disassemble XYZ1111
** No association **	Construct XYZ1234
	Dismantle XYZ1234
	Construct XYZ5678
	Dismantle XYZ5678
	Construct XYZ1111

5.8 EPISTLE engineers' intuitive concerns

What the engineers noticed was that ECM v2.22's installation-facility pattern does not work at the 'big' level of oil rigs.

The notion of installing a physical oil rig seems to be meaningless – with no physical process to mark it. One could argue that it is reasonable to install a physical pump in a pump facility – as the facility could have a number of pumps. But it seems intuitively odd – and superfluous – to have to install the physical oil rig in the oil rig facility.

Installation seems to presume some difference, maybe only a possible difference, between what is being installed and what it is installed into. But this is not true for the physical oil rig and the oil rig facility. For example, the idea of deinstalling the physical oil rig and installing a new one just does not make sense. At some level, installation implies possible deinstallation and multiple installations – and this is not possible here.

Similarly the PF101's assemble association with ORF111 seems to be different from the assemble association the physical pump's components have with it. We can imagine changing a pump's piston, we take the old one out, put it aside and install the new one. But it does not make sense to say that we are replacing an oil rig's pump facility with a new facility. How could we take a pump facility out and put it to one side – while installing a new one?

The workaround solution was to ignore the physical version – and stick with just a logical oil rig. The success of this practice seems to prove that there is no pressing practical need for more than one oil rig.

While the practice 'worked', it left the engineers with all the intuitive concerns about logical things raised earlier in relation to pump facilities - exacerbated by there being nothing to 'physicalise' the logical oil rig – as there was with pumps. For example, given that we can see and touch an oil rig, does this have to be the logical oil rig as there is no physical oil rig?

These seem to indicate that the physical/logical distinction and the assembly and installation association patterns are not the right 'shape' for this part of reality – for what the ECM v2.22 convention considers the 'big' level.

Rather than complicating things with a different pattern for the 'big' level - what the BORO analysis reveals is a single pattern that spans the levels, without the need for workarounds.

6 Re-engineering the extended domain

We now start the BORO re-engineering of the extended ontology, following a similar process to that of the first analysis; we start with the individuals – in the next section – and then – in the following section – analyse the upper levels.

6.1 Re-engineering the ground level

We work from the ground up – we start here in this section with the individuals; the physical and logical oil rigs. And then in the next section we move onto their associations and associated activities.

6.1.1 Re-engineering the new individuals

Most people have a reasonably clear idea of what a physical oil rig is.

The physical and logical oil rig

But few would claim they have a clear an idea of what an oil rig facility is or how a physical oil rig is installed in it. However, applying Locke's test brings sufficient clarity to see what is going on. This reveals that both the beginning and end of ORF111 and XYZ1111 are the same. Oil rig facility ORF111 becomes available when the physical oil rig XYZ1111 is constructed and is not longer available after XYZ1111 is dismantled. Furthermore ORF111 follows XYZ1111 around – occupying the same place at the same time. We cannot imagine a situation where we can see and touch one and not the other. This is quite different from the situation with PF101 and XYZ1234. These do not share beginnings or endings. And they only share places for parts of their lives: there are times when they are clearly in different places.

This gives ORF111 and XYZ1111 the same spatio-temporal extension. Under the criterion of identity for individuals, this makes them the same thing. Let's agree to call this constructed oil rig XYZ1111. This analysis backs up the users of ECM v2.22 who only committed to a single oil rig. Unfortunately for those users, under ECM v2.22 rules, it had to be a logical oil rig facility – rather than the more intuitively comfortable physical oil rig.

This re-engineering makes the installation association between ORF111 and XYZ1111 (and its associated activities) superfluous. There is no need to link the two as they are the same thing. It also explains the unusual characteristics of this installation association (raised in *§5.8* - above) where we cannot imagine the installed physical oil rig being5.5(a)-5(ins(a)-3.8(i)-3.5(l)1.(led)5.s(a)-3.5(n)-1.d4(newed
o
e

6.1.2 Re-engineering the new assembly associations

The demise of ORF111 and its installation association leaves us with only three unre-engineered associations in the extended ECM v2.22 model. These are all assembly associations:

- Constructed oil rig XYZ1111 (originally ORF111) assembly component pump PF101
- Constructed oil rig XYZ1111 assembly constructed pump XYZ1234
- Constructed oil rig XYZ1111 assembly constructed XYZ5678

Whether these assembly associations are 'eternal' or 'temporary' – or, in other words, time-less or time-relative – makes a difference to how we re-engineer them. The first assembly association is 'eternal' and time-less – PF101 is always part of XYZ1111. The second and third assembly associations are 'temporary' and time-relative – XYZ1234 and XYZ5678 are only assembled into XYZ1111 for a part of their lives. The re-engineering reveals the different natures of the relations.

The 'dependent' component relation

This assembly relation is more than timeless – it also marks that PF101 is dependent upon XYZ1111, the oil rig of which it is part. We intuitively know why we call PF101 dependent. It is an integral, essential, component part of the oil rig. It is impossible to imagine this component pump (as a component) without the oil rig of which it is part. In other words, if the oil rig did not exist, its component pump could not.

When the existence of one thing (the component pump) depends upon another (the oil rig), philosophers call this an ontic dependency[26]. In this case, the component relation marks the ontic dependency. It is this ontically dependent component relationship that marks PF101 as a component pump and the lack of it that

[26] We need to consider PF101 as spatio-temporally dependent upon XYZ1111 – rather than dependent upon XYZ1111 at any point in time in which it exists. This saves us from getting bogged down in considering exceptional circumstances where things might not be sufficiently clear cut. We may also want to say that it is, in general, true that at any time which PF101 exists, XYZ1111 will also exist – this is not implied by the notion of dependency. And if we do say this, we have to explain the exceptional cases. For example, where an engine is all that survives of an aircraft after an accident. Here the investigators might still say 'this is aircraft A's engine'. We could reply that we would also say 'this is all that is left of the aircraft' implying the aircraft still exists. Also my intuition is that if the engine was then 'installed' in another aircraft, we would then say that 'this *was* aircraft A's engine'. But the details of this analysis do not affect the dependency relation.

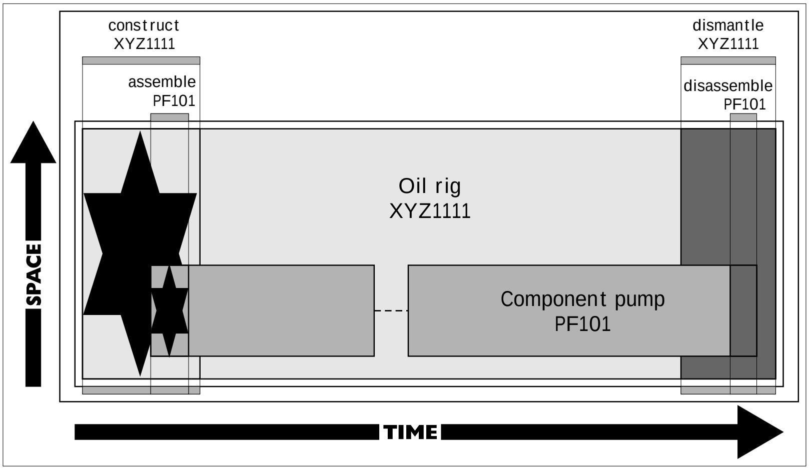
marks pumps XYZ1234 and XYZ5678 as constructed. (Things are a bit more subtle than this suggests – under some circumstances, a constructed pump can, accidentally (contingently), have a dependency relation, but under no circumstances can a component pump not have one.)

But what is the nature of the component relationship? It becomes clearer if we look at the spatio-temporal relationship between the two individuals involved in the association: the constructed oil rig and its component pump. Simplifying things a little, let's assume that the constructed pump XYZ1234 is installed as the oil rig is constructed – and that XYZ5678 is deinstalled as the oil rig is dismantled.

If we then analyse PF101 and XYZ1111's beginnings and endings we see that they happen at roughly the same times and in roughly the same places. The difference being PF101's start assemble activity is only a part (both spatially and temporally) of the XYZ1111's overall construct activity.

Furthermore, if we consider their whole extents, we see PF101 follows XYZ1111 around – at any particular time being a pump-shaped part of it. We can represent this in a space-time map – as in *Figure MC1–35* below.

Figure MC1–35 The oil rig and its component pump space-time map



If the pump had been installed after most of the oil rig was constructed or deinstalled before the oil rig was finally dismantled, then the same general relationships would hold – but the details of the picture would be slightly less simple.

Figure MC1–35 makes one thing very clear, that PF101 is a spatial part of the oil rig for most, if not all, of the rig's life – it is consistently part of the oil rig. This makes sense. A component will typically be consistently part of its whole.

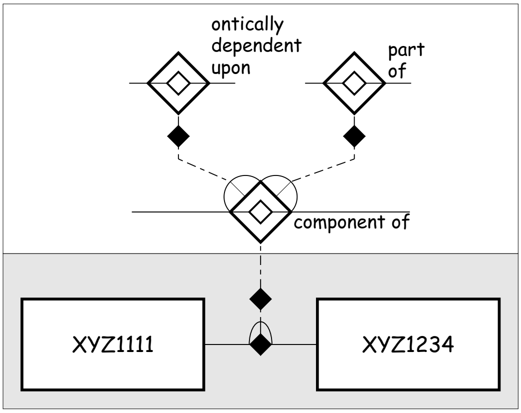
This explanation sheds light on why the oil rig does not have a component hierarchy that goes down much below the level of pump, while the physical pumps have a component hierarchy that goes much lower.

When the designer specifies the oil rig, he or she specifies the pump in terms of what it is required to do. It may well be that many different types of pumps, could adequately meet the requirements – for example, a centrifugal or a screw pump. There is no reason why someone cannot swap a pump of one type for a pump of the other type at some stage of the oil rig's life. This means we can be confident that the centrifuge is a component of the centrifugal pump and the screw of the screw pump, but we cannot be confident that either of these will be components of the oil rig – spatial parts for most of its life.

The 'eternal' XYZ1111 assembly PF101 association

In BORO the 'eternal' assembly association is realised as a tuple and its associated relation class (it is not realised as an individual, as, for example, the temporary installation associations were). We re-engineer the assembly association instance into the couple <PF101, XYZ1111> and its association into the relation class 'component of'. This relation class is a sub-class of both the 'ontically dependent on' and 'whole-part' relation class. A model of this part of the ontology is given in *Figure MC1–36* below.

Figure MC1–36 XYZ111 assembly PF101 object schema



The constructed pump assembly associations

The re-engineering of the two ‘temporary’, time-relative, associations:

- Constructed oil rig XYZ1111 assembly constructed pump XYZ1234
- Constructed oil rig XYZ1111 assembly constructed XYZ5678

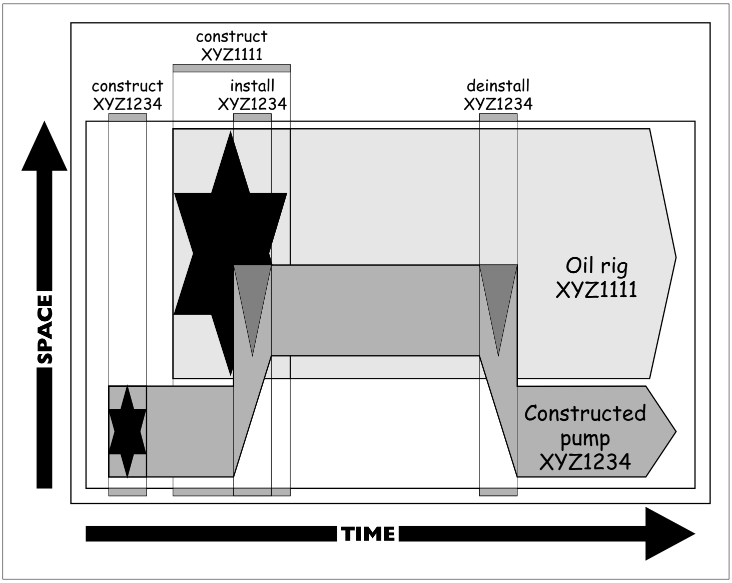
follows a similar pattern to the re-engineering of the installation associations in the initial analysis. This revealed them as individuals rather than relations. But in this case the analysis also reveals them as superfluous.

We consider the XYZ1111 assembly XYZ1234 association first and then apply an analogous re-engineering process to the XYZ1111 assembly XYZ5678 association.

The XYZ1111 assembly XYZ1234 association

As we now have a picture of the spatio-temporal relationship between XYZ1111 and PF101 (see *Figure MC1–35*) and the spatio-temporal relationship between PF101 and XYZ1234 (see *Figure MC1–8*), we can easily construct one for XYZ1111 and XYZ1234. *Figure MC1–37* below shows the result.

Figure MC1–37
XYZ1111 and
XYZ1234
space-time
maps



The spatio-temporal relationship between XYZ1111 and XYZ1234 in *Figure MC1–37* and that between XYZ1111 and PF101 in *Figure MC1–35* both of which have – in the ECM model – assembly associations illustrate the two senses of assembly in ECM v2.22.

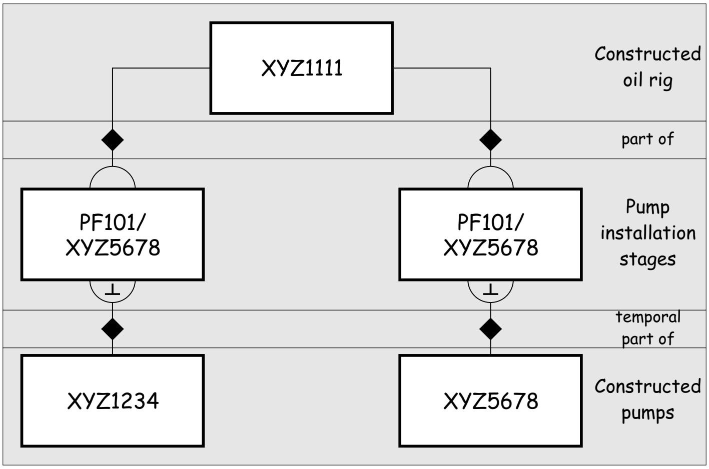
In the first sense the items being assembled have an 'eternal' timeless relationship. Where, for example, PF101 is an 'eternal' part of XYZ1111. In spatio-temporal terms, PF101 is a spatio-temporal part of XYZ1111. In other words, there is no time at which any (spatial) part of PF101 is not also part of XYZ1111.

In the second sense, the items being assembled only have a temporary relationship. Where, for example, XYZ1234 is a 'temporary' part of XYZ1111. In spatio-temporal terms, a temporal part of XYZ1234 (the installation stage PF101/ XYZ1234) is also a part of XYZ1111. The other temporal parts of XYZ1234 are not part of XYZ1111. In space and time terms, at some times no (spatial) part of XYZ1234 is part of XYZ1111 (at that time) – at other times all of XYZ1234 (at that time) is part of XYZ1111 (at that time). This relationship is better described from a spatio-temporal viewpoint as 'temporary inclusion' rather than part of.

Taking the re-engineering of the installation associations (in *§.The nature of the installation associations* - in *§4.1.3*), we can see this as another manifestation of installation stages. The constructed pump has installation stages as 'eternal' temporal parts. The installation stages are also 'eternal' temporal parts of PF101 which is in turn an 'eternal' part of XYZ1111. Then, as parthood is transitive, the installation stages are also 'eternal' parts of XYZ1111. This gives us the analogue of the assembly association's link to XYZ1111 – but an 'eternal' rather than a 'temporary' association.

Figure MC1–38 below shows a model of the two re-engineered assembly associations – using installation stages.

Figure MC1–38 XYZ1111, installation stages and constructed pumps object schema



As the schema makes clear, the direct 'part of' relations between the installation stages and the constructed oil rig, XYZ1111, are superfluous. They are composed of the 'eternal' part of relations between the installation stages and PF101 and another eternal part of relation between PF101 and XYZ1111.

6.1.3 Re-engineering the new activities

The re-engineering has winnowed down the individuals in the ontology. The oil rig facility and its installation association have disappeared – as has, for different reasons, XYZ1111's constructed pump assembly associations. All this has a follow on effect on the activities linked to these associations. Of the eighteen activities listed in *Table MC1–2*, the six in *Table MC1–3* have been dropped or replaced as a result of BORO analysis.

Table MC1–3 The dropped/replaced activities

Association	Activity	Status
XYZ1111 installation ORF111	XYZ1111 install PF111	Association dropped
	XYZ1111 deinstall PF111	Association dropped
XYZ1234 assembly XYZ1111	XYZ1234 assemble XYZ1111	Association replaced
	XYZ1234 disassemble XYZ1111	Association replaced
XYZ5678 assembly XYZ1111	XYZ5678 assemble XYZ1111	Association replaced
	XYZ5678 disassemble XYZ1111	Association replaced

Furthermore, the eight installation and construction activities (listed in *Table MC1–4*, below) were re-engineered in the original analysis.

Table MC1–4 The original re-engineered activities

Association	Activity
XYZ1234 installation PF101	XYZ1234 install PF101
	XYZ1234 deinstall PF101
XYZ5678 installation PF101	XYZ5678 install PF101
	XYZ5678 deinstall PF101
** No association **	Construct XYZ1234
	Dismantle XYZ1234
	Construct XYZ5678
	Dismantle XYZ5678

Which only leaves these four activities (listed in *Figure MC1–5*) to be re-engineered in this extended analysis.

Table MC1–5 The remaining activities

Association	Activity	Status
PF101 assembly XYZ1111	PF101 assemble XYZ1111	XYZ1111 replaces ORF111
	PF101 disassemble XYZ1111	XYZ1111 replaces ORF111
** No association **	Construct XYZ1111	
	Dismantle XYZ1111	

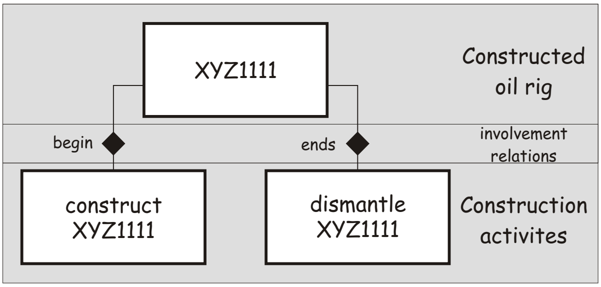
Assembly activities

We have names and ECM v2.22 models for these assembly activities, but what are they in reality? In ECM v2.22 these assembly activities 'begin' and 'end' the assembly associations[27], which only last for the intervening period. For installation associations that were re-engineered into installation stages, it made sense to keep these activities to begin and end the stages. But the assembly associations have been re-engineered into 'eternal' relations, for which it makes less sense to talk about beginnings and endings. These relations are 'eternal' because, in a sense, they have no beginning or end. So it looks like there is no job for these assembly activities in BORO, that, like ORF111 they can be dropped.

Construction activities

The re-engineering of the constructed oil rig's construction activities is no different from that of the constructed pumps (see *§4.1.5*). A temporal part of the oil rig is involved in each construction activity (see analogous *Figure MC1–11* and *Figure MC1–14*) – but it is simpler to just consider the involvement relations that mark the beginning and ending of the oil rig. These are shown in *Figure MC1–39* below.

Figure MC1–39 Oil rig XYZ1111's construction activities

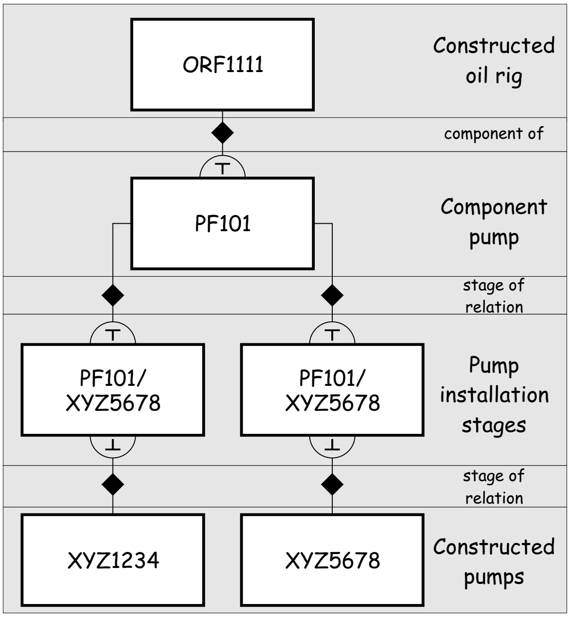


6.1.4 BORO's new ground level ontology

We now have a decent picture of the new ground level of individuals (and their inter-relationships) – and this is diagrammed in *Figure MC1–40* below (ignoring activities). To make the model even simpler only the immediate part of relationships are represented.

[27] [7] "Conceptually, the start and end points of the lifetime of an association are defined by the activities that bring the association about and terminate it." §4.3 Association, *EPISTLE Framework V2.22*.

Figure MC1–40 The new extended ground level ontology



There are some new major changes at this level – easily seen by comparing *Figure MC1–40* above with the earlier BORO *Figure MC1–16* and ECM v2.22 *Figure MC1–34*:

- The logical oil rig facility has been collapsed into the physical/constructed oil rig – and its installation association and its associated activities dropped.
- The facility assembly activities have been collapsed into the initial and final installation activities.
- The physical pumps' assembly associations have been subsumed in the installation stages 'part of' relations.

Hopefully these illustrate the trend in BORO analyses towards a simpler, slimmed down, ontology – where things in the original system either disappear (such as the oil rig facility's installation association) or are collapsed together (such as the physical oil rig and the oil rig facility).

6.2 Re-engineering the extended upper ontology

We now move onto re-engineering the upper ontology. This will give us a clearer picture of the generality of the patterns – component of relations and life history types. It will also illustrate how a clearer more general picture of these patterns subsumes the lower level patterns leading to a (structurally) simpler ontology.

6.2.1 A caveat

This analysis is only intended to illustrate the process. Within the space of a case study it is not possible to walk through a full analysis. (For example, we have only space for a sketchy characterisation of some of the classes in terms of a few individuals.) But the space is sufficient to make a plausible analysis and show the general direction it is going. This provides some direct experience of how simple general patterns emerge.

6.2.2 Re-engineering to a natural level of generality

The ontology here is quite interconnected so we need to tease out the various strands. A key part of any BORO analysis is to try and capture the general elements of a pattern – to let it rise to its natural level of generality. Typically this results in a simpler, more sophisticated, picture.

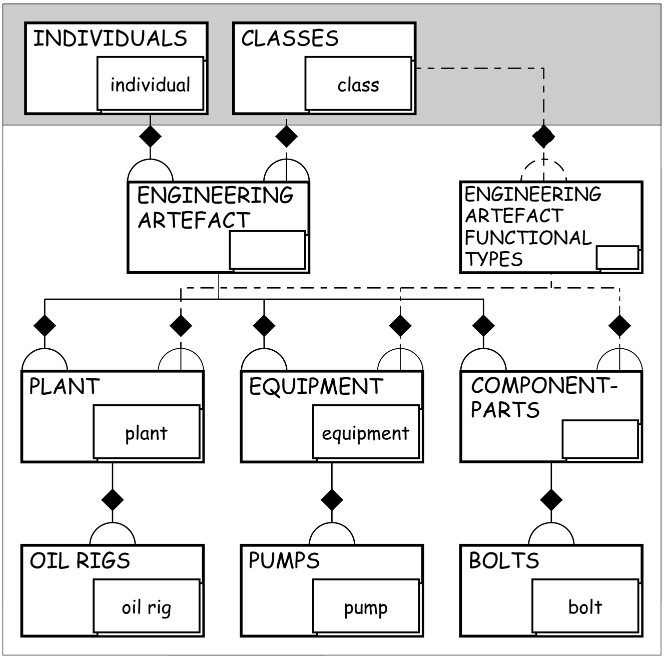
In the original engineering, we came across a distinction between centrifugal and screw pumps that worked orthogonally to the life history pattern. We also referred to an EPISTLE distinction between 'big' (plant), 'medium' (equipment) and small (component-part) things. We firstly generalise this.

Generalising functional dimension

Firstly we need to recognise that plant, equipment and component-parts are all engineering artefacts; in other words, are parts of the bigger 'engineering artefacts' class.

We noted earlier that what differentiates a centrifugal and a screw pump is the way in which they work – using a centrifuge or a screw to pump. We can argue – at least for the purposes of the case study[28] – that there is a similar distinction between plant, equipment and component-parts as types of engineering artefact. Then we can regard distinctions as differentiating between functional types – as shown in *Figure MC1–41* below.

Figure MC1–41 Engineering artefact functional types object schema



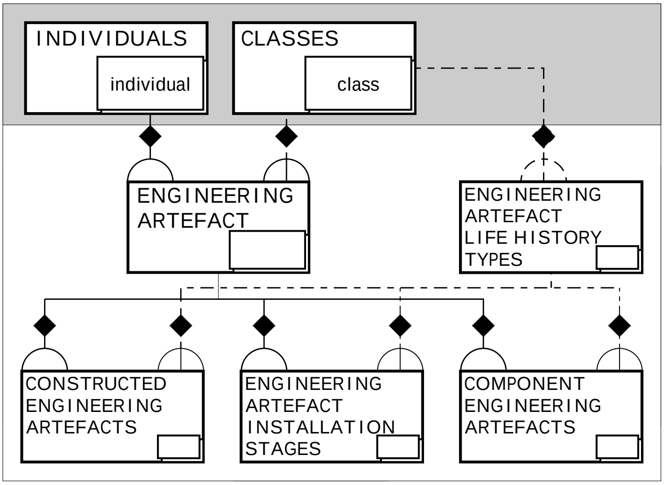
For the purposes of this example, the function types dimension has been explicitly recognised in the form of an engineering artefact functional types class (of classes).

Generalising life history types

The generalisation of functional types provides a good basis for the generalisation of the other pattern in our original analysis: life history types. There seems no reason not to allow engineering artefacts to have the full range of life history types (and so their associated activities) – as shown in the model in *Figure MC1–42* below.

[28] This is a simplified version of the categories of engineering artefact. For example, engineers often have levels between plant and equipment – where, for example, equipment is part of a unit of operation, which is in turn part of a plant. The distinctions here are probably not fundamental - just rough rules of thumb. However as this is not the focus of our analysis, we stick with the simplification.

Figure MC1–42 Engineering artefact life history types object schema


INDIVIDUALS
individual
CLASSES
class
ENGINEERING ARTEFACT
ENGINEERING ARTEFACT LIFE HISTORY TYPES
CONSTRUCTED ENGINEERING ARTEFACTS
ENGINEERING ARTEFACT INSTALLATION STAGES
COMPONENT ENGINEERING ARTEFACTS

Now that we have generalised the notion of life history types up to the level of engineering artefacts, the life history types in our earlier analysis (pump – *Figure MC1–20* and equipment – *Figure MC1–21*) are all subsumed under this one types class – and no longer required.

Aspects of identification criteria

These generalisations (functional and life history types) can be seen characterising aspects of our normal identification criteria (for more on identification criteria see *Appendix B*'s *Identity (and 'identification') criteria*). Types at the level of constructed or component pump have, for us, identification criteria: we can reasonably reliably identify instances of them. This is not true of a life history or functional type. By itself, it does not teel us enough to identify its instances. However, once we know both types, both aspects, the identification criteria crystallises and we are in a position to identify its instances.

Generalising the 'component of' relation

The extended domain included a 'component of' relation (assembly association) pattern that, within ECM v2.22 had practical restrictions on its application. The BORO framework does not suffer from these restrictions, and so we can use the relation to guide a generalisation across 'big', 'medium' (equipment) and 'small' things to the level of engineering artefact.

The preceding analysis revealed a single individual physical oil rig, XYZ1111, which has PF101 as a component – in other words, they are related by a component of relation. This can be generalised. The obvious first step is to recognise the component of relations between component pumps and constructed oil rigs.

But the scope for generalisation does not stop there, the component of relation is not restricted to pumps and their oil rigs – pumps are not the only kinds of components or oil rigs the only kind of things with components. For example, at the individual level, oil rig XYZ1111 has lots of other types of component equipment – for example, air conditioners. This suggests that there can be component equipment in general.

Similarly, at the individual level, constructed pump XYZ1234 could be installed in a range of plant other than an oil rig - for example, a ship. This suggests as a first step we need to recognise a more general class, plant, and take 'component of' as a relationship between it and component equipment.

The next step is to determine whether there can be component plant – as well as constructed ones? And can component equipment be components of both component and constructed plant? It would seem so. For example, while an oil rig is big, it is not so big that it cannot, in principle, be a component of something bigger. And in practice there are examples where it is, such as large oil field exploited by a configured[29] group of oil rigs – where each individual oil rig is a component of the whole group. In this case there are component and constructed oil rigs; where the constructed oil rigs are installed and can be replaced, though we may not make much of the constructed and component distinction where the oil rig is unlikely to be replaced. (It is worth noting that the practical problems of constructing and organising something so big probably makes it unlikely that the configured group of oil rigs is, in turn, a component of an even bigger complex of groups.)

And if there can be component plants, can these have equipment components? If the configured group of oil rigs is designed as a whole, there will be a component

[29] It is important for the identity of the dependent oil rig that there is a configuration. See ships and fleet example in *§6.3.4* .

pump that is a component of the component oil rig. And, in general, can components have components? There seems in principle no reason why not, though, in practice, it may not make sense to have a deep component hierarchy (see *§.The ‘dependent’ component relation* in *§6.1.2*).

The next question is whether and hop24.6‘oooe’op24.6patt6.4(tirough, i031.0791 -1.366

Engineering artefacts' two dimensions

Life history types are one dimension of engineering artefacts – telling us about their temporal shape. Functional types can be seen as another - telling us about their spatial configuration – and so how they might be used. The two dimensions are broadly[30] orthogonal. Orthogonal dimensions (even broadly orthogonal ones), as ECM v2.22 (and Aristotle[31]) recognise, are useful for their combinatorial expressive power. For example, many classes can now be derived as combinations of the members of the two dimensions - the class constructed pump can be derived from the classes pump and constructed engineering artefact, and so is, in at least one sense, superfluous.

6.2.3 Architecting dimensions

An important factor to be considered, from an information economics point of view, when choosing which dimensions to use, is the extent of overlapping and disjointness within and across dimensions. Too little overlapping (too much disjointness) restricts fruitfulness (one of BORO's goals) – and too much overlapping (too little disjointness) can lead to chaos. This is because overlapping/disjointness determines the range of multiple classification (how many of which classes an object can be member of) and so the potential for generalisation (for more details on these economic aspects see *Business Objects: Re-engineering for re-use*[32]).

ECM v2.22 and BORO provide examples of two different strategies for architecting the dimensions. ECM v2.22 works by a mixture of intuition and stipulation: whereas BORO has a more principled and empirical approach.

One of ECM v2.22's great strengths is its clear stipulation of overlapping / disjointness – and its balanced application of it. It quite clearly and simply specifies that subtypes in its four top level dimensions must overlap, while those on differ-

[30] Where 'broadly' implies that there might be some dependencies between the subtypes of the dimensions that exclude certain combinations of functional type and life history type. A broad or rough orthogonality is sufficient to give us the combinatorial power we need. Note: EPISTLE orthogonality is also broad in this sense. Further analysis would specify the shape of the dependencies between the 'orthogonal' dimensions.

[31] Aristotle's categories (substance, quality, etc.) are structurally broadly orthogonal dimensions.

[32] *Business Objects: Re-engineering for re-use* , Chapter 6, Section 3, particularly Section 3.2

ent branches of the same dimension cannot[33]. ECM v2.22 does not provide any general reasons why a framework should have this structure – though it has good arguments for individual cases. But there seem to be two factors driving this:

Firstly, the need to bolster (often unreliable) intuition and experience as a basis for determining disjointness and overlapping, and

Secondly, the need to take advantage of combinatorial combination.

BORO has what appears to be a similar top-level structure – with three mutually exclusive and exhaustive categories. However, these categories are of a purely formal nature, more describing the tools we are going to use to classify the world, than describing different types of thing in the world. Below this level there are no pre-specified constraints.

This enables there to be a wider variety of structures. Sub-types can be introduced without having to be part of a dimension – and dimensions can therefore be introduced as needed. And when 'dimensions' are introduced, whether their sub-types overlap on occasion or not needs to be worked out – typically the assessment involves the identity conditions of its members[34]. This is neatly illustrated in its treatment of the life history type dimension – discussed above. It is tempting to assume that the life history types, and any activities associated with them, are disjoint – or even just stipulate that they are. But the analysis reveals that it is morefruitful to recognise how they can overlap.

6.2.4 Life history dimension's disjointness

We start by considering the mereological possibility space – what kinds of disjointness and overlapping are possible. Then the analysis follows a simple pattern. For each of the spaces, the possibility of something existing in it is considered. Overlapping is demonstrated by describing an individual that meets the identification criteria associated with the two (apparently disjoint) classes – so must

[33] This restriction means that within of the top level dimensions, the sub-types form a tree rather than a lattice. It makes each sub-division of a dimension's sub-types into another mutually exclusive sub-dimension.

[34] This process is no more than that of characterising the mereology of the class's extension described in *§4.2.1*.

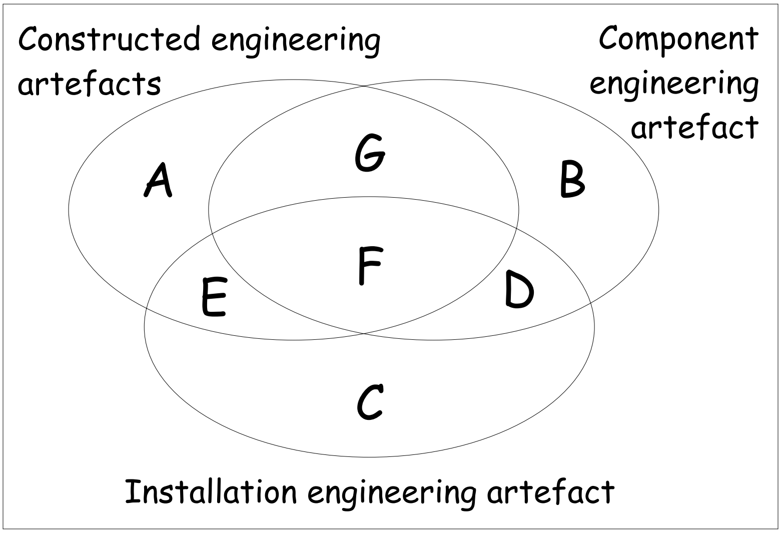
belong to both classes – so the classes overlap. This reveals mereological patterns that are more sophisticated than ECM v2.22's constraints allows – and that many classes that are unthinkingly thought of as disjoint in fact overlap.

Classes' mereological possibility space

Characterising the mereological possibility space of two classes (such as constructed and dependant pump) is simple. There are four possibilities: one is part of the other or vice versa or they overlap or they do not.

But we have three life history types – and characterising their mereological possibility space offers more possibilities. Not only can we ask whether all three classes have an overlap, but also whether each pair of classes overlap. This does not exhaust the possible mereological relations of the classes. For example, a pair of classes may overlap without overlapping a third class, as illustrated by areas E, F and G in the Venn diagram in *Figure MC1–44* below. In general, we can determine what questions to ask by considering the possibilities of overlap as areas within a space. *Figure MC1–44* represents this space for the life history types - with the areas marked by letters. For each overlap space, we determine whether it is occupied by trying to provide an individual that 'lives' there – or explaining why the space is empty.

Figure MC1–44 Engineering artefact life history types' mereological space


Constructed engineering artefacts
Component engineering artefact
A
G
B
F
E
D
C
Installation engineering artefact

Distinct – not sub-classes

The first and most important question is whether the life history types classes are sub-classes (parts) of one another. To determine that they are not, we need to find a member of each class that is not a member of any of the other classes. In other words, we need to find examples of members that live in the spaces named A, B and C in *Figure MC1–44*.

Our domain has already provided us with these. XYZ1234 is an example of A, it is a constructed pump but not a component pump or a pump installation stage – and so to the corresponding engineering artefact classes. Similarly PF101 is an example of B and PF101/XYZ1234 an example of C. As these areas are occupied, the classes are distinct – none of them are subclasses of any of the others. The results of this analysis is shown in *Table MC1–6* below.

Table MC1–6 Life history types mereology – disjointness

Area	A	B	C
Constructed engineering artefacts	X		
Component engineering artefacts		X	
Engineering artefact installation stages			X
Occupied	Yes	Yes	Yes

Identifying overlapping

Sometimes the answers to these mereological questions can be surprising. In the example, it might seem that the three life history types are completely disjoint – that they do not overlap. However, this turns out not to be so.

Consider a new version of our example in which XYZ568 is not installed: where the component pump PF101-D has only the constructed pump XYZ1234-D installed, and this is constructed elsewhere prior to installation. Applying John Locke's test the component pump and the sole pump installation stage share their beginnings and endings – as well as the life in between - and so are co-extensive. Whereas the individual pump has a different beginning – prior to installation and so is not co-extensive. This component pump / installation stage lives in area D[35].

[35] The analysis can be complicated with questions of modality. The co-extensiveness is contingent, another pump may have been installed . In this case what we would call PF101-D (but which has a different extension) would not be identical with PF101/XYZ1234-D. BORO's position is based upon the spatio-temporal extensions, which are identical.

Consider another new version of our example where an engineering artefact, such as a storage tank, XYZ1234-E, is constructed as SF101-E in the oil rig. Storage tanks, particularly big ones, are typically built (welded) in situ. After a time the tank develops a serious fault, is dismantled in situ and replaced with XYZ5678-E. Here XYZ1234-E, the original constructed storage tank, and its installation stage, SF101/XYZ1234-E share the same beginning and ending as well as the life in between and so are co-extensive. But the component storage tank SF101-E has a different ending and so is different. The constructed / installation stage storage tank lives in area E.

Storage tanks[36] are a good example as they are typically constructed in situ. It is more difficult to provide a convincing example for things such as pumps that are not. Furthermore, we tend to think of the pump being assembled of parts and then subsequently connected to the plant, whether this is done in situ or not. However the storage tank example shows that there is no constraints on the notions of constructed and installation stage that preclude them being co-extensive. Presumably with sufficient ingenuity and the right motivation someone could actually work out a way of constructing a pump in situ that also naturally seems like an installation.

Consider yet another new version of the example where the constructed storage tank XYZ1234-F is again constructed in situ as SF101-F in the oil rig, but this time is only disassembled when the oil rig is dismantled. As SF101-F has only one constructed storage tank, XYZ1234-F, installed, it only has one installation stage, SF101/XYZ1234-F. We apply John Locke's test – checking whether they have the same beginning and ending. They do. The construction of the constructed storage tank is also the installation of the component storage tank and the beginning of the installation stage. The dismantling of the constructed storage tank is also the de-installation of the component storage tank and the ending of the installation stage. And they all share the same life in between. So they all have the same spatio-temporal extent (are co-extensive) and so, by the

[36] Storage tanks are an example raised by the EPISTLE team of where their stipulated disjointness of installation and construction break down.

criterion of identity for individuals, they are same individual. This constructed / component / installation stage storage tank lives in area F.

These last two cases (E and F) have some interesting implications for the identity of life histories' begin and end activities. Within ECM v2.22, activities of a different type are necessarily disjoint (by fiat). But in the case of F, the construct activity that begins the constructed storage tank, the install activity that begins the storage tank installation stage (association) and the first installation activity that begins the component storage tank all seem to be the same.

It would seem to be stretching credulity to say that the constructed storage tank was constructed in situ and then after this the component storage tank was installed (even if only a second later) – accompanied by no physical activity. The same reasoning applies to the end activities – the deinstall and dismantle activities of the pump is the same event. Without the ECM v2.22 fiat, there is no reason why we cannot recognise them as identical.

The G area needs a different approach as it is unoccupied. Instead of looking for an occupant, we need to explain why it cannot be occupied. It is enough to show that if something is both a constructed and a component artefact, then it must also be an installation stage. Consider a constructed, component artefact, it either has one or more installation stages. If it only has one installation stage, then there is no difference between the installation stage and the component – and so they are identical. If a component has two or more installation stages, it seems impossible for it to remain constructed. The nub of the argument is that when a constructed artefact is deinstalled, the component ceases to exist – until the next artefact is installed – but the constructed artefact does not[37]. At that moment their spatio-temporal extensions diverge. So if something is both constructed and a component, it must have one installation stage – and these must all be identical.

[37] This does turn on an assumption that the separation of a constructed artefact's parts does not lead to its ending its existence. If one regards separations and re-assemblies as marking the boundaries of intermittent periods of non-existence, then one could create an example for G. It would involve a co-extensive separation and deinstall followed by a co-extensive re-assembly and install – of the same component-constructed artefact – where the two installation stages involve the same constructed artefact.

The results of this second analysis is shown in *Table MC1–7* below:

Table MC1–7 Life history classes' mereological space

Area	D	E	F	G
Constructed engineering artefact	X		X	X
Component engineering artefact		X	X	X
Pump installation engineering artefact	X	X	X	
Occupied	Yes	Yes	Yes	Yes

6.2.5 The nature of a component's dependence

A component artefact has both a component of relation – and a life history that begins with an initial install and ends with a final deinstall. Why do these necessarily go hand in hand? The reason is that the initial install of a component artefact does not happen in a vacuum – it is installed as a component of something. Furthermore, the component's deinstall must happen either before or as part of the composing artefact's dismantling. These two aspects of the life history's shape ensure that whenever the component exists, the composing whole also does. You cannot have the component without its composing whole. The shape of the component life history enforces the dependency relation – and the relation is the result of the shape of the life history.

6.2.6 Co-extensive life history activities

What enables the different types of event to overlap, leading to co-extensive artefacts is that they are characterised in orthogonal ways. The construction activity involves, typically, assembling the elements of the artefact. The install activity involves setting up the artefact in situ (in the right place in relation to the artefact it is a component of). If the setting up is done by assembling the elements (as in cases E and F above), then the individual activity is both an install and a construct activity – and the artefacts they begin could, potentially, be identical.

6.2.7 The temporal nature of these life histories

In this analysis we have only considered, as candidatesfor components of these types of life history, activities that mark the artefact's temporal boundaries. For example, a constructed artefact has two temporal boundaries, its beginning marked by a construct activity and its ending marked by a dismantle activity. Whereas a component life history includes interim install and deinstall activities as these mark the beginnings and endings of stages of its intermittent existence. These life history types only broadly characterise the possible shapes of life histories – but they do so sufficiently well to meet our goal of distinguishing between component and constructed artefacts.

6.2.8 Managing the abundant combinatorially 'possible' types

Even at this level of broad characterisation, the life history types presented here do not represent all the possible combinations. Why can't there be an artefact that has install as its begin activity and dismantle as its end. In other words, one that starts like a component with an install, but continues as the first constructed artefact installed until this is dismantled. These activities perfectly consistently mark out a spatio-temporal extension – that of the constructed artefact minus the extent before its first installation.

However, there is no evidence of anyone actually using this new install/dismantle artefact – nor does it seem likely to have any practical use. This and other 'possible but impractical' histories are excluded by restricting the life history types to those that are useful.

This abundance of possibilities occurs at all levels of the ontology. The characterisation[38] of individuals as spatio-temporal extensions could be extremely promiscuous, leading to an abundance of individuals. The art of a good business ontology is to winnow out the ones that are of no use. This is typically done in stages at different levels. In this example, the general artefacts class starts the

[38] Quine famously made a similar characterisation in Word and Object (1960 - p. 171) – "a physical object comprises simply the content, however hetereogeneous, of some portion of space-time, however disconnected or gerrymandered."

process of winnowing out the useless, impractical ones – and this is continued by the (slightly) less general life history types.

6.3 Generalising the re-engineered patterns

Until now we focussed our attention on engineering artefacts and components. The natural general level of the patterns revealed by the analysis is much wider. It covers more than engineering artefacts and components – and with a suitable twist can be extended even further.

6.3.1 Wider than engineering artefacts

The component pattern applies to a vast array of things. Positions in organisations are an obvious case. The President of the United States is a component of the Government of the United States. The MegaBank's Accounts Department is a component of MegaBank. My heart (another type of pump) – and other organs – are components of my body: this has become practically relevant with the advent of organ transplants – where the constructed organ can be installed in multiple components. Note that, from a life history perspective, our organs are much like storage tanks, they begin with a combined construct and install – and usually end with a combined dismantle and deinstall. Also, much like pump facility for designers and operators, we are more interested in the component that the constructed organ.

6.3.2 Wider than component artefacts

The last example provides a case that involves two kinds of dependence. My heart is a component of my body – and is owned by me. The ownership relation is an ontically dependent relation: if my heart 'belonged' to someone else would it be mine? My (component) heart necessarily belongs to me. This gives rise to the familiar pattern of component, installation stages and constructed. If I have a heart transplant, my heart component has two installation stages. The first involves the (constructed) heart I was born with, the second someone else's (constructed) heart. A less surgical, more natural example is a snake and its one component skin and many constructed skins.

There are less biological examples. We talk about ‘my car’ as a component whose life spans the various cars that are ‘installed’ – as when we say “my company car is now a Sierra, it was an Escort”. We talk about ‘my desk’ – where the component desk remains the same, even if we are hot-desking and so at a different (constructed) desk every day.

6.3.3 Mass dependents

The pattern can be generalised to masses as well as bodies. A river is a water component dependent upon the riverbed. The petrol in my car’s tank is a petrol component – and so on. Though in this case the nature of masses makes the notion of installation stages less meaningful. Except in the cases where we can easily temporally divide the masses, such as when rivers regularly run dry.

6.3.4 Unconfigured component-less stages

The component pumps in the case study had a specific configuration in the oil rig. The dependent component of relation was for a specific place (position) in the oil rig’s configuration. This place/position is what enabled us to make sense of a component that persists through a deinstall and install. There are cases where the ‘components’ have no configuration. Then, depending on your point of view, there are no components or the components collapse into the installation stages.

A good example of this would be the ships in a fleet. Here ships may join/leave (be de/installed in) the fleet. Fleets (as with other similar types of collections – herds, flocks, etc.) usually have a loose structure, where the various ships typically have no specific position to act as a ‘home’ for subsequent installation stages. So there is no way to link a specific deinstall with a subsequent install in the life history of a ‘component’ of the fleet.

It is worth noting that ECM v2.22’s installation association depends for its existence on its facility and physical object – so the framework cannot be generalised to cover situations where there is no configuration in which to house the facility.

6.4 The re-engineered domain

This last stage of the re-engineering revealed the simple life history dimension implicit in the constructed and component patterns. And also shown how generally it can be applied. This is a typical result of a BORO analysis – which has been repeated over a range of domains.

7 The Case Study

This case study illustrates both how the BORO approach works and the benefits it brings. ECM v2.22 provides a good example of how difficult it is, using just intuition and experience, to get a simple, general framework that does not lead to counter-intuitive results. The BORO re-engineering shows that the application of ontological principles to the model brings simple general patterns without the counter-intuitive results.

BORO

APPENDICIES

Appendix A

The BORO Program and its Approach

Appendix B

Core BORO concepts

Appendix C

EPISTLE

Appendix D

The EPISTLE Core Model

APPENDIX A

THE BORO PROGRAM AND ITS APPROACH

The BORO Program is being established to help businesses reap the benefits of applying a business ontology approach to the business levels of systems integration and development.

A.1 What does the business ontology approach do?

It enables the building of a standard model of business understanding (a reference ontology model) that is at least as rich as that embedded in current systems – probably much richer. And that is significantly (structurally) simpler than the combined systems.

And this reference ontology has all sorts of applications in system integration and development – for example:

> It can act as the basis for a lingua franca for communication between disparate systems, substantially simplifying their rich business level integration.
>
> It can act as the basis for a business level specification in the development of systems, effectively eliminating one costly step in the system development process – and substantially simplifying subsequent steps.

The way this works is that the approach provides a sophisticated framework that:

Firstly, enables the extraction of the business understanding embedded in current systems to be extracted and integrated into a common, objective, standard and modelled in a form that is easily understandable by both humans and computers.

Secondly, enables this understanding to be modelled in far simpler structures (patterns) than those existing in the current systems.

An overview of core concepts, including that of an ontology is in *Appendix B—Core BORO concepts*.

A.2 How can businesses reap the benefits?

Businesses can reap the benefits from this approach in a number of ways. They can actively participate in the program, working with the BORO team to extract the business knowledge from their current systems (typically re-engineering a legacy system) and merging this into the Reference Ontology Model – taking advantage of the work already done and contributing towards a better model. Or more passively, they can work on their own, building upon the results of the work to date. They can use the BORO approach to re-engineer their own legacy systems and/or the use the BORO model as a starting point for development and integration.

A. 2.1 Developing awareness and experience

It is recognised that an important part of gaining acceptance for a business ontology based approach will be developing awareness and experience of it. As part of this BORO will:

develop learning materials that would help businesses understand what a business ontology is and why they could benefit from modelling it. (This case study is part of the learning materials.)

work with businesses, helping them to use the approach and harness the benefits of doing so. (This case study is based upon an example of this.)

A. 2.2 The need for an explanation

As with all 'new' approaches some work needs to be done explaining what they are and why they are useful. For example, the BORO approach relies heavily upon the technical notion of an identity criterion -with which most people are unfamiliar. The full explanation of this and other core concepts, including ontology, is given in *Appendix B—Core BORO concepts*. And the case study illustrates how these work in practice.

Some work also needs to be done in explaining why these principles are useful. The case study answers this point directly by example. It shows how using the principles give a firm foundation for an improved analysis of what the individual *pump facility PF101* is and so what a general *pump facility* is. It will also show how the simple pattern that this analysis reveals, is one that that appears again and again in different guises in a wide variety of situations. Thus providing an opportunity to consolidate many (apparently) different applications under a few simple general patterns.

Appendix A

APPENDIX B

CORE BORO CONCEPTS

The concepts behind the BORO approach will be relatively new to many IT people, and the explanation below is intended give them a sufficient idea of its core concepts to enable this case study to be understood. It should also help people familiar with the concepts to understand the approach BORO is taking. If you are interested in a more detailed description, this can be found in *Business Objects: Re-engineering for re-use*.

BORO is built on the core concept of a business ontology – where the starting point for an ontology is the categories of object that exist. This is discussed below. Identity and grounding are concepts central to the key principles we are looking at here – these are explained in the section on categories of object.

B.1 Business ontology

We start by looking at what Ontology(-with-a-capital-O) and ontology(-with-a-small-o) are.

B.1.1 Ontology(-with-a-capital-O)

Ontology(-with-a-capital-O) is an ancient philosophical discipline which can be traced back to the Ancient Greeks. It is a branch of metaphysics and its subject matter is existence and its nature.

Famously Quine[39] claimed that the problem of Ontology can be stated in three words 'What is there?' – and the answer in one 'everything'. Not only that, but Quine says "everyone will accept this answer as true." However he accepts that "there remains room for disagreement over cases."

Doing Ontology usually involves, at some stage, developing at least a part of an ontology(-with-a-small-o). And doing Business Ontology(-with-a-capital-O) involves developing a business ontology(-with-a-small-o).

B.1.2 ontology(-with-a-small-o)

Central to an ontology(-with-a-small-o) is an inventory of the types of object that (can) exist and a categorisation of this list, often by the types of existence they (can) have. (So a business ontology(-with-a-small-o) will include a categorisation of the business things that (can) exist and the types of existence they (can) have.) It is perhaps easier to understand what this means by starting with the notion of ontic commitment.

Ontic commitment

By nature, information is about something. More precisely, any system of information (whether a business computer system or a scientific theory) refers to things – and so implies that they exist. These things are the information's ontic commitment.

For example, consider the specification for an oil rig that includes plans for a pump facility with the tag no. (name) PF101. This tag no. ontically commits the specification to the existence of PF101. As the specification also uses the term 'pump facility', it can be regarded as committing to the existence of a general pump facility[40].

[39] In W.V. Quine, 1948 - 'On what there is', Review of Metaphysics, Vol. II, No. 5, reprinted in Quine, 1961, 'From a Logical Point of View', 2nd edition (New York, Harper & Row).

[40] Of course, it is possible to take a nominalist position and regard the general term pump facility in the specification as denoting a multitude of individual pump facilities rather than referring to a single general pump facility. The BORO approach does not take this position – and, to keep the description simple, I follow the approach.

The specification will also contain details of various types of equipment and how they are related – for example, what is connected to what. These details ontically commit the specification to a whole range of objects.

[As an aside, this is why the preferred BORO approach is to start with an existing business system, which has, however mangled, an ontic commitment. Working out the ontic commitment of a blank sheet of paper, which is sometimes recommended as a starting point for systems development, is not a serious option.]

A general framework

We can tease out even more general commitments. It is likely that the specification makes an almost tacit distinction between general things, such as the general pump facility and individual things, such as PF101. There may be, for example, standard symbols for general things such as pump facilities. Then there may be a standard symbols, such as tag nos., for individual things, such as the individual pump facility PF101. We can and should recognise the categories of general things and individual things[41] as part of its ontic commitment.

We start to reveal an ontology when we determine the general types of objects that exist – of which the categories general and individual things are just one example – and how these are inter-related. To provide a complete ontology we need to also provide some explanation of what these types of things are – and the types of existence they have. Together these form what might be called the general ontological framework.

B.1.3 Why have a general framework?

At first blush, ontic commitment may not seem very radical or indeed useful. Currently many people using a business system assume that they have a clear idea of its ontic commitment – at least at the not-too-general level of pump. Similarly many IT people assume that their systems are a good reflection of these kinds of things – that the information in the system is a map of its ontic commitment.

[41] These 'categories' have been rationalised in a number of ways – e.g. universal and particular or property and individual – each with its own baggage.

Many people also do not see the point of fitting a general ontological framework over these. But it turns out that there is a point. Among other things, the general framework reveals our current notions of ontic commitment are not as good as we think and it helps us to make them better.

A clear idea of the ontic commitment

This comes as a surprise to most people. They (we?) understandably think that experts have a reasonably clear and consistent idea of their ontic commitment. But it becomes quite obvious that they do not when they try to fit it into a general framework.

They certainly know what the words they use refer to, and prove this by correctly acting upon and issuing instructions using them. They can also provide a kind of model of the not-too-general things they are committing to. But when they try to fit these consistently into more general commitments, they typically run into serious difficulties.

For example, experienced engineers have a sufficiently good idea of what a pump facility is to do their jobs. They have proved again and again that they can design and maintain one. But when they try to fit their commitment to pump facilities within a general framework they run into difficulties. It turns out to be very difficult for them to do this on their own in a satisfactory way.

Turning accuracy up a notch or two

Perhaps we should not be so surprised, as something similar happens during the automation of manual systems. The description of what the system currently does or should do by the people working the manual system is rarely if ever adequate for an automated system (though it is a good starting point).

The shift from manual to automated systems creates a requirement for more consistent and accurate models of the business. The shift to a general ontological framework takes the requirement for consistency and accuracy up a notch or two. One reason people have difficulties is that they are (not yet) used to working at these more demanding levels.

B.1.4 An 'objective' reference ontology

Ontic commitment lays the foundation for a reference ontology – one which can be used as an 'objective' standard, constructed of particular applications - by focusing on the things in the world. Different specifications may model pump facility PF101 in different ways – or focus on different aspects – but in some sense they must all commit to the existence of the *same* PF101.

The need for a general framework

But what we are looking for is a general ontology covering more than just one pump facility. What we want is an ontology to act as a common reference point across the full range of businesses – a reference ontology. However, as the scope of an ontology expands, there is more scope for inconsistency and so the importance of a consistent general framework increases. When we have a 'big' ontology (certainly by the time we get to a reference ontology) we need to be sure that the commitments we are making are consistent across its full range. To do this we need to add a general framework to our ontic commitment.

B.1.5 Developing a reference ontology

How are we going to develop the reference ontology? Most people have difficulty in modelling the essential general framework and a plausible reason for this is that they do not have one to start with. And constructing one is nothing like as simple as it might seem – involving, among other things, a demanding degree of accuracy. This makes a build-our-own strategy unattractive.

Taking the general framework

Luckily Ontology(-with-a-capital-O) provides us with another option. Since the days of the Ancient Greeks, it has taken as one of its major tasks as identifying the major different kinds of thing (and different kinds of existence things have) and fitting them into a coherent framework – in other words, building a general framework.

An important part of this task has been developing a sufficiently accurate understanding of what such a framework is and how it fits together. In particular, much time has been devoted to understanding the issues that the framework needs to address and how these relate to one another.

Ontological relativity

It turns out that there are a group of closely inter-related central issues that face anyone trying to build a general ontological framework. Different groups of philosophers, motivated by different concerns, have developed a range of frameworks - each proposing its own set of inter-related solutions to the central issues, each with its own characteristics. What they have found is that their proposed solution to any one of the issues has profoundly influenced how they can approach the others.

This situation can be characterised as ontological relativity – to highlight that there is (at least currently) no single absolute ontology; more a series of intimately inter-connected ontological options.

Tailoring a reference ontology

Ontological relativity adds another layer of difficulty to building a reference ontology – we cannot just select the single standard ontological framework, we have to choose one. How should we choose? This is where the business ontology approach differentiates itself. It tailors the framework best suited to BORO's purposes. One that:

is suitable for the types of things (and their relationships) that business systems typically commit to. In particular, that provides the right kind of solutions to the range of central issues that a business systems' ontological framework is likely to encounter.

encourages the subsumption of a range of not-so-general patterns under simple general patterns.

Certainly the first of these (and to some extent the second) are not the usual concerns that motivated the professional philosophers who work in Ontology. Nevertheless what I find really surprising is that most – if not all - of the apparently (philosophically) technical central issues they discuss have direct practical analogues in modelling a business's ontology. Perhaps this is why it is not too much work to tailor an approach that suits our purposes.

Sources for the framework

The source for the framework is work done in a branch of analytic philosophy that starts with Gottlieb Frege, and takes in Rudolf Carnap and W.V.O. Quine and more recently David Lewis and Mark Heller. For example, the notion of ontic commitment used above to introduce ontology was developed by W.V.O. Quine.

The kind of central positions absorbed into the ontological framework include:

- Individuals are four-dimensional extensions.
- Universals (general things) are classes.
- Possibility is described in terms of possible worlds.

If you want to find out more about how these influence BORO's ontological framework, look in my book *Business Objects: Re-engineering for re-use*. There is also a good philosophical introduction to the overall position in Mark Heller's *The ontology of physical objects: four dimensional hunks of matter*.

B.1.6 Aspects of the business ontology approach

Business ontology's practical focus not only dictates its choice of framework, it also influences the approach to developing the full reference ontology – differentiating it from the more academic approach taken by philosophers. For example:

- In general, the scope of a philosopher's analysis will be determined by his particular interests. Whereas the scope of business ontology is dictated by the ontic commitment of business systems.
- Philosophers are often only really interested in the general framework and so restrict their analysis to that – based upon some well tried examples. Businesses need a *deep* reference ontology that includes most things that are committed to by a number of applications – both not-so-general and individual. For example, they will need an engineering section of the ontology that includes not-so-general things, such as 'pump facility' and an international banking section that includes individuals, such as 'The Bank of England'.
- Philosophers can admit defeat, saying that in some areas they do not have a solution that meets their high standards. Business ontology does not have this luxury – it has to deliver a *comprehensive* reference ontology. While it is useful to be able to recognise when a solution is not up to a 'high' standard, if it is 'good enough' and all there is, then the reference ontology will need to go with it - recognising that there is scope for improvement.

- Philosophers prime mechanism for judging their ontologies is peer review. A business reference ontology is primarily subject to a different kind of quality control. Its litmus test is whether it works in practice. It needs to be useful in integrating/developing computer systems. The better it works, the more likely it will be used. And there is also a natural limit on the depths to which quality can sink, because a reference ontology has to be *industrial strength*. If is does not work well enough, it will not be used.

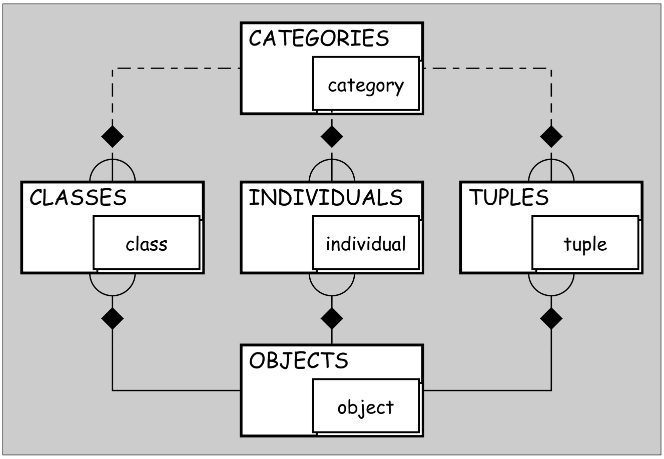
B.2 Ontological Categories of Object

As said earlier, central to an ontology(-with-a-small-o) is a notion of what types of thing exist and the types of existence they have. This notion is embedded in the ontological framework. Traditionally the top-most levels of the ontological framework are known as ontological categories and in BORO, as in some other systems, these classify the different kinds of existence that things can have.

Within BORO everything that exists is classified into one of three mutually exclusive categories called (as shown in *Figure MC1–0.1* below):

- Individuals,
- Classes, and
- Tuples

Figure MC1–0.1 BORO'S Top-most Level Object Schema



Individuals (also called particulars) are spatio-temporal extensions. These are things that are extended in space and time - typically objects that we can see and touch; like the chair I am sitting on or the pen on my desk.

Classes are collections of objects. They can be quite small, such as the pens in the cup on my desk, of very large, pens in general. What characterises them is that they have members. The class of pens has each individual pens, including the pen on my desk, as members.

Tuples are part of the ontological apparatus that we use to handle relations. A tuple is a sequence of objects. Following mathematical conventions the name of the tuple of A followed by B is written as <A, B>, where A and B are names of objects.

There are a number of structural points that are worth noting.

> Every object belongs to one and only one of these types. It is clear from their descriptions that the types are disjoint.
>
> Any object (whether individual, class or tuple) can be a member of a class or a position in a tuple sequence.
>
> From a meta-framework point of view, the three categories and the general type objects are classes with instances of the types as members. So, all objects are members of the class objects, all individuals are members of the class individuals.

This is a very brief sketch, if you want more details see the book *Business Objects: Re-engineering for re-use*.

B. 2.1 Identity (and 'identification') criteria

Within Ontology, identity is a key concept as illustrated by its catch-phrase 'no entity without identity'. What this means is that we cannot (or, at least, should not) claim an entity exists unless we have some idea of what its identity criterion is.

An identity criterion can be seen as a way of characterising the nature of something. It can also be seen as a principle or rule for determining whether, when we make different identifications (often linked to names or descriptions), we are talking about the same (or different) things. It will often be phrased as such a principle - " If x's [rule], then they are the same".

Within BORO's business ontology, identity criteria are given for the three categories of object and apply to instances of these categories. The brief descriptions of these categories above give us a clue as to what their criteria are.

If individuals have the same spatio-temporal extension, then they are the same. In less technical jargon, if two things are always in the same place at the same time, then they are the same. A classic example is the two names 'Morning Star' and the 'Evening Star'. Ancient astronomers at first thought these were two different planets. However as their observations became better, they realised that these were in the same places at the same times – that they were one thing, the planet Venus.

If classes have the same members then they are the same. This is not always trivial. For example, the class of equiangular triangles and the class of equilateral triangles have the same members – and so are the same class.

If tuples are composed of the same objects, in the same sequence, then they are the same. So <A, B> and <A, B> name the same tuple. Whereas <A, B> and <B, A> do not – as the objects are in different sequences.

Only the three categories have identity criteria (for their members) – and they only have one identity criterion each. As the categories are disjoint, this means every object (every member of the three categories) has one and only one identity criterion. It would be complicated to have more as we would then need to show that they could not, in principle, conflict.

If we want to identify an object, then we need something other than the identity criteria. Typically an object will have all sorts of relations both with itself and other objects. In different contexts, many of these can be a basis for identifying the object. Within a particular system, there may be an identification criterion – a rule for identifying the object. Often there will be a number of identification criteria. In general, there is no reason why one of these criteria should be ontologically superior to the others. Identification criteria are not priviledged in the way identity criteria are.

What one can say (loosely) is that within the sum of an object's properties there needs to be some way of identifying the object. Otherwise it would not be clear what the object is. Notice that this sets no upper limit on the number of different identification criteria an object may have. And in practice there are normally quite a few – some more reliable, some less reliable.

So, for example, the table at which I am sitting is an individual with a spatio-temporal extent. I need to give a reasonable

tlisptariicu

[illegible]

This is because when we experience the world, we typically perceive particular individuals. We perceive a particular horse, not the general class of horses. When two people want to make sure that they are talking about the same horse, they can go

APPENDIX C

EPISTLE

Most of this material has been drawn from the EPISTLE Home Page, which can be found at http://www.stepcom.ncl.ac.uk/epistle/epistle.htm.

C.1 EPISTLE and its ECM

The European Process Industry STEP Technical Liaison Executive (EPISTLE) arose directly out of the need to handle the potential overlap and duplication of effort between projects and organisations developing standards for the exchange of technical information for the Process Industries using STEP.

STEP (ISO 10303) is an emerging international standard designed to facilitate the exchange of computer-based “product models”. The focus of the work on STEP is the interchange of information about the product itself, rather than the drawings used to represent it. More details on STEP can be found on its home page - http://www.nist.gov/sc4/www/stepdocs.htm.

C.2 Objectives

The two main purposes of European Process Industry STEP Technical Liaison Executive (EPISTLE) were defined as:

- To identify potential for collaboration between parties involved in developing standards for the exchange of technical information,
- To organise and deliver shared solutions to these problems.

C.3 Membership

The initial European members of EPISTLE were ATLAS, Caesar Offshore, PISTEP, POSC, ProcessBase and SPI-NL. There is now growing involvement from organisations outside Europe including NIST, PISA, ShipSTEP, PDXI and Oil rigSTEP.

C.4 EPISTLE's ontological flavour

EPISTLE has a clearly structured and described overall framework that reflects the business (rather than the system that is being implemented) and explicitly describes this framework in its model.

This is because it takes its goals, which have a strong ontological flavour, seriously. It clearly states as its first 'fundamental principle' that:

> "Our principal concern is to represent the underlying nature of the things that we are modelling …"

And adds that it does not want to:

> "… represent as structural elements of our model constraints that may not be appropriate under all circumstances or that may change with the course of time or are of an organisation-specific nature."

C.5 EPISTLE and BORO

This strong ontological flavour makes it, from an ontological point of view, ahead of most other mainstream models (though, until relatively recently it had no exposure to business ontology). As a good part of the ontological groundwork had been done, work can start directly on the application of the BORO principles. It has, for example, something remarkably similar to an ontological framework. So there is no need to build a framework from scratch, and it is easy to show the improvements as changes in this framework.

EPISTLE has also been a good partner. It has put a significant amount of effort into the BORO analysis – leading to good, strong results. Once EPISTLE became aware of business ontology, it recognised its potential for the ECM and was – and is - working hard on applying it.

BORO

Appendix C

APPENDIX D

THE EPISTLE CORE MODEL

The EPISTLE Core Model (ECM) is a general model that is used as a basis for developing consistent application specific models. It is perhaps easiest to let EPISTLE itself explain its overall structure. Below is an extract from Section 2.3 'The Generic Entity Framework' in the *EPISTLE Framework V2.22* document.

> "*Things* (including characteristics, activities and associations) are defined in terms of a *generic entity type* based upon the underlying nature of the entity (*thing*). Within the *generic entity framework*, a *thing* is anything that exists, real or imagined. The generic entity type serves to place the *thing* in a universal context that is constructed of the particular role being played by it.

The Framework defines four sets of subtypes or concepts representing different aspects of a *thing*[42] :-

- *subject* the underlying essence of the thing (*object, association, activity*, *characteristic* and subtypes thereof)
- *instantiation* whether it is a *class*, a *specific* instance or a *typical* instance
- *life cycle* its life cycle state - *actual, predicted, required* or *planned*
- *reality* *real* or *imaginary*

[42] This structure has several structural similarities to (and differences from) the 'original' ontological framework – the Aristotelian categories. In both of these a thing typically belongs to one and only one of each of the categories. Aristotle has a privileged category called substance, which deals with essences – EPISTLE's subject category works in a similar way. Aristotle then has a number of less privileged categories called attributes – EPISTLE less priviledged categories are called qualifiers. However there are significant differences in content. For example, Aristotle classified relation as an attribute, whereas EPISTLE has it as a subject. EPISTLE classifies reality and instantiation as qualifiers, whereas Aristotle deals with them outside the framework.

A fully defined *generic entity type* is defined by choosing a *subject* subtype, an *instantiation* subtype, a *life cycle* subtype and a *reality* subtype. …

The four sets of subtypes are said to be orthogonal to each other in that any one thing can only be placed in precisely one entity type in each of the four sets.

The *instantiation*, *life cycle* and *reality* subtypes are known collectively as *qualifiers* as they are used to qualify the underlying essence of the thing that is defined by the *subject* subtype.

In general, "relationships" between things are represented as *association* entities."

Figure MC1–0.1 below is taken from the same section. It shows the four 'dimensions'[43] and their main subtypes.

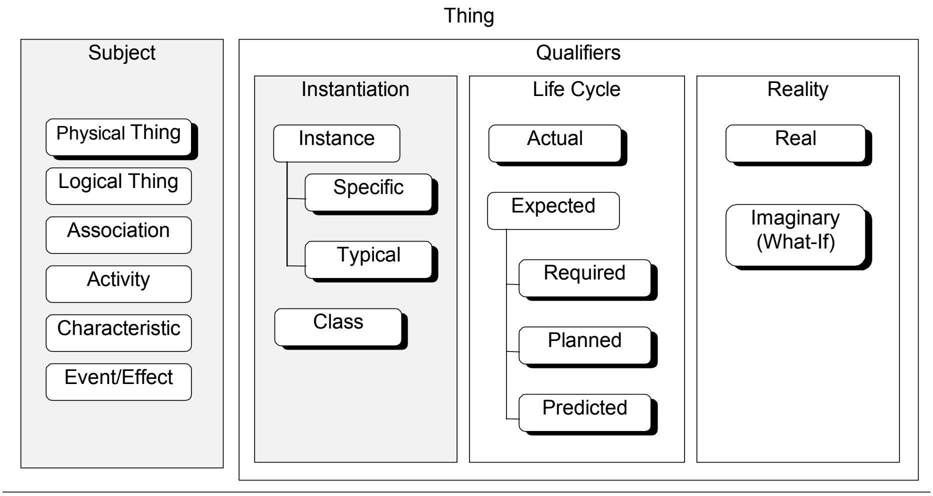


Figure MC1–0.1 Overview of the EPISTLE Core Model

[43] The use of dimensions to classify information is now becoming an active research area. It is building upon work done in Library Science originated by S.R. Ranganathan in the 1930's with the Colon Classification, where he introduced the name 'facets' for dimensions. Further details can be found in Ranganathan, S.R. Colon Classification, Basic Classification. 6th ed. New York: Asia Publishing House, 1963. The ideas were taken up in IT – see, for example, Ruben Prieto-Diaz's proposal for "faceted classification" of a reusable software library – based on the concepts he found in library science. [R. Prieto-Diaz and P. Freeman., Classifying Software for Reusability. IEEE Software, 4(1): 6-16, January 1987.]

MCI - Bibliography

Home Pages

These home pages contain useful information.

[A]The EPISTLE Home Page. This can be found at:

http://www.stepcom.ncl.ac.uk/epistle/epistle.htm.

It contains both historic and current versions of the EPISTLE Core Model (ECM), as well as some of the EPISTLE related documents listed below.

[B]The STEP Home Page. This can be found at:

http://www.nist.gov/sc4/www/stepdocs.htm

Documents

[1] *EPISTLE Framework V2.22(Version 2.0, Issue 1.22)*, Editors Chris Angus and Peter Dziulka, 10 Jul 1998. Available from the EPISTLE Website.

The various ISO STEP standards:

[2] *ISO 10303-221: Product Data Representation and Exchange-Part 221. Functional data and schematic representation for process oil rig*.

[3] *ISO 10303-227: Product Data Representation and Exchange-Part 227*. Oil rig spatial configuration.

Other relevant ISO standards

[4] *ISO 1087, Terminology – Vocabulary*.

[5] The Life of a Pump

Graphical Notation I
BG1— *Constructing Signs for Business Objects*
Graphical Notation II
BG2— *Constructing Signs for Business Objects' Patterns*

Volume - M
M—The BORO Re-Engineering Methodology

Book - MO
MO—The BORO Re-Engineering Methodology: Overview

MO1—*The BORO Approach to Re-Engineering Ontologies*

Book - MW
MW—The BORO Methodology: Worked Examples

Worked Example 1
MW1—*Re-Engineering Country*
Worked Example 2
MW2—*Re-Engineering Region*
Worked Example 3
MW3— *Re-Engineering Bank Address*
Worked Example 4
MW4—*Re-Engineering Time*

Book - MA
MA—The BORO Re-Engineering Methodology: Applications

MA1—*Starting a Re-Engineering Project*
MA2—*Using Business Objects to Re-engineer the Business*

Book - MC
MC—The BORO Re-Engineering Methodology: Case Histories

Case History 1
MC1—*What is Pump Facility PF101?*

INDEX

E–T